GPC® R/T94
General Purpose Controller
Relays or Transistors; 9 Inputs, 4 Outputs

TECHNICAL MANUAL
Intelligent module ABACO® Block Serie M; Size 22.5x82x90 mm; Container for Ω rails type DIN 46277-1 and DIN 46277-3; CPU 89C2051 or 89C4051 with 14MHz quartz; 2K or 4K Bytes FLASH and 128 Bytes SRAM; Code compatible with family 51 µP; Real Time Clock with 256 Bytes internal SRAM; Back Up circuitry for RTC and SRAM, through Lithium battery; Serial E² up to 1K Bytes; 9 galvanically isolated NPN input lines; Galvanically isolated external INT signal; 4 output lines: 5A relays in version R, Optocoupled 4A, 45 Vdc NPN Darlington without heat sink and with back EMF protection diode, in version T; LEDs to visualize the status of all the I/O signals; I/O signals connection through quick release connectors; 2 TTL I/O signals; 16 bits Timer Counter register; A/D Converter with 11 bits of resolution, conversion time 60 ms, analog input signal in the range 0÷10Vdc, 0÷20 mA or 4÷20 mA; Serial line in TTL, RS 232, RS 422, RS 485 or Current Loop; Possibility of Idle Mode or Power Down Mode; High efficiency switching power supply on board; Supply of galvanically isolated section: +24 Vdc; On board logic power supply: 5 Vdc or 10÷40 Vdc or 8÷24Vac; Protection against transients by TransZorb™; Telecontrol Firmware featuring ALB (ABACO® Link BUS) communication protocol; Wide range of development software available like C Compiler, Assembler, BASIC Compiler BASCOM 8051, HTC-51, etc.
DOCUMENTATION COPYRIGHT BY grifo®, ALL RIGHTS RESERVED

No part of this document may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, either electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without the prior written consent of grifo®.

IMPORTANT

Although all the information contained herein have been carefully verified, grifo® assumes no responsibility for errors that might appear in this document, or for damage to things or persons resulting from technical errors, omission and improper use of this manual and of the related software and hardware.

gifo® reserves the right to change the contents and form of this document, as well as the features and specification of its products at any time, without prior notice, to obtain always the best product.

For specific informations on the components mounted on the card, please refer to the Data Book of the builder or second sources.

SYMBOLS DESCRIPTION

In the manual could appear the following symbols:

⚠️ Attention: Generic danger

⚡️ Attention: High voltage

Trade Marks

GPC®, grifo®: are trade marks of grifo®.

Other Product and Company names listed, are trade marks of their respective companies.
## GENERAL INDEX

**INTRODUCTION** ................................................................................................................... 1

**CARD VERSION** ................................................................................................................... 1

**GENERAL INFORMATIONS** .................................................................................................... 2

- **CPU** .................................................................................................................................. 4
- **POWER SUPPLY SECTION** .................................................................................................. 4
- **CLOCK** ............................................................................................................................... 6
- **MEMORY DEVICES** ............................................................................................................. 6
- **GPC® R94 OUTPUT SECTION** ............................................................................................. 6
- **GPC® T94 OUTPUT SECTION** ............................................................................................. 6
- **INPUT SECTION** .................................................................................................................. 7
- **FIRMWARE ALB X94** ........................................................................................................... 7
- **A/D CONVERTER AND TTL I/O LINES** ............................................................................ 7
- **SERIAL COMMUNICATION SELECTION** .......................................................................... 7

**TECHNICAL FEATURES OF GPC® R94** .................................................................................. 8

- **GENERAL FEATURES OF GPC® R94** ............................................................................. 8
- **PHYSICAL FEATURES OF GPC® R94** ............................................................................. 8
- **ELECTRIC FEATURES OF GPC® R94** ............................................................................. 10

**TECHNICAL FEATURES OF GPC® T94** .................................................................................. 11

- **GENERAL FEATURES OF GPC® T94** ............................................................................. 11
- **PHYSICAL FEATURES OF GPC® T94** ............................................................................. 11
- **ELECTRIC FEATURES OF GPC® T94** ............................................................................. 12

**INSTALLATION** ....................................................................................................................... 14

- **CONNECTIONS** .................................................................................................................. 14
  - **CN4 - POWER SUPPLY CONNECTOR** ............................................................................ 14
  - **CN7 - CONNECTOR FOR OPTOCOUPLED INPUTS** .................................................. 24
  - **CN8 - GPC® R94 RELAY OUTPUTS CONNECTOR** ..................................................... 26
  - **CN8 - GPC® T94 TRANSISTOR OUTPUTS CONNECTOR** ........................................... 28
  - **CN6 - A/D CONVERTER INPUT CONNECTOR AND TTL I/O LINES** .................. 30
- **LEDS** .................................................................................................................................... 32
- **INTERRUPTS** ....................................................................................................................... 32
- **BACK UP** ............................................................................................................................ 32
  - **I/O CONNECTION** ........................................................................................................... 34
- **ANALOG INPUTS SELECTION** ........................................................................................ 35
- **POWER SUPPLY SECTION** ............................................................................................ 35
- **JUMPERS** ............................................................................................................................ 36
  - **3 AND 5 PINS JUMPERS** ............................................................................................... 36
  - **2 PINS JUMPERS** ........................................................................................................... 38
- **MEMORY SELECTION** ......................................................................................................... 38
- **SERIAL COMMUNICATION SELECTION** ........................................................................ 40
SOFTWARE

PERIPHERAL DEVICES SOFTWARE DESCRIPTION ............................................................... 43
  SERIAL EEPROM ........................................................................................................... 44
  BACKED SRAM + SERIAL RTC ...................................................................................... 44
  DIGITAL INPUTS ......................................................................................................... 45
  DIGITAL OUTPUTS ....................................................................................................... 47
  JUMPER J1 (RUN/SETUP) - A/D LINE ....................................................................... 47
  CPU PERIPHERALS ..................................................................................................... 47
  ALB (ABACO® LINK BUS) SOFTWARE DESCRIPTION ............................................... 50
WORKING MODE SELECTION ......................................................................................... 50
SETUP MODE .................................................................................................................. 50
RUN MODE ..................................................................................................................... 53
GENERAL PURPOSE COMMANDS ................................................................................ 54
  MASTER RESET ........................................................................................................... 54
  READ FIRMWARE VERSION ........................................................................................ 54
  CHECK EEPROM READY FOR WRITING ..................................................................... 54
  PRESENCE BYTE OUTPUT .......................................................................................... 55
  PRESENCE BYTE INPUT .............................................................................................. 55
  AUX SIGNAL CONFIGURATION SETTING .................................................................... 56
  AUX SIGNAL CONFIGURATION ACQUISITION ......................................................... 56
  READ BOARD CODE ................................................................................................ 56
DIGITAL I/O PORT BYTE MANAGEMENT COMMANDS .................................................... 57
  OUTPUT PORT SET .................................................................................................... 57
  INPUT PORT ACQUISITION ......................................................................................... 57
DIGITAL I/O PORT BIT MANAGEMENT COMMANDS ....................................................... 58
  OUTPUT BIT SET ....................................................................................................... 58
  TIMED OUTPUT BIT SET ............................................................................................ 58
  OUTPUT BIT CLEAR .................................................................................................... 59
  TIMED OUTPUT BIT CLEAR ....................................................................................... 59
  INPUT BIT OR AUX INPUT ACQUISITION ............................................................... 60
  DEBOUNCED READ OF AN INPUT PORT BIT ....................................................... 60
  SQUARE WAVE STARTING WITH "1" OUTPUT BIT .................................................. 61
  SQUARE WAVE STARTING WITH "0" OUTPUT BIT .................................................. 62
  SQUARE WAVE OUTPUT BIT .................................................................................... 63
16 BIT COUNTER MANAGEMENT COMMANDS .............................................................. 64
  16 BIT COUNTER READ ............................................................................................ 64
  16 BIT COUNTER RESET .......................................................................................... 64
MESSAGES MANAGEMENT COMMANDS ........................................................................ 65
  LAST MEMORIZABLE MESSAGE ACQUISITION ...................................................... 65
  STORING A MESSAGE ............................................................................................... 65
  READING A MESSAGE ............................................................................................... 66
SRAM+RTC MANAGEMENT COMMANDS ....................................................................... 67
  CLOCK SETTING ....................................................................................................... 67
  CLOCK READ ........................................................................................................... 68
  RTC SRAM WRITE .................................................................................................... 68
  RTC SRAM READ ..................................................................................................... 68
1-WIRE® COMMUNICATION BUS LINES ........................................................................... 69
COMMANDS TO MANAGE THE 1-WIRE® BUS LINES .................................................. 70
RESET 1-WIRE® BUS AND GET PRESENCE PULSE .................................................. 70
WRITE A BIT TO 1-WIRE® BUS ............................................................................. 70
READ A BIT FROM 1-WIRE® BUS .......................................................................... 71
WRITE N BYTES TO 1-WIRE® BUS ...................................................................... 71
READ ROM CODE FROM 1-WIRE® BUS ................................................................. 72
MATCH ROM COMMAND ON 1-WIRE® BUS ......................................................... 73
SKIP ROM COMMAND ON 1-WIRE® BUS .............................................................. 73
ALARM SERCH COMMAND ON 1-WIRE® BUS ...................................................... 74

9 BITS MASTER-SLAVE COMMUNICATION MODE ............................................. 75

BIBLIOGRAPHY ........................................................................................................ 78

APPENDIX A: COMMANDS OF FIRMWARE ALB X98 ........................................... A-1

APPENDIX B: ON BOARD COMPONENTS DESCRIPTION ..................................... B-1

APPENDIX C: ALPHABETICAL INDEX ................................................................. C-1
FIGURES INDEX

FIGURE 1: GPC® R 94 BLOCK DIAGRAM ................................................................. 3
FIGURE 2: GPC® T 94 BLOCK DIAGRAM ............................................................... 5
FIGURE 3: COMPONENTS MAP GPC® R94 (COMPONENT SIDE) ...................... 9
FIGURE 4: COMPONENTS MAP GPC® R94 (SOLDER SIDE) .............................. 9
FIGURE 5: COMPONENTS MAP GPC® T94 (COMPONENT SIDE) ..................... 13
FIGURE 6: COMPONENTS MAP GPC® T94 (SOLDER SIDE) ............................ 13
FIGURE 7: CN4 - POWER SUPPLY CONNECTOR .................................................. 14
FIGURE 8: LEDS, CONNECTORS, ETC. LOCATION ON GPC® R94 ................... 15
FIGURE 9: LEDS, CONNECTORS, ETC. LOCATION ON GPC® T94 ................... 15
FIGURE 10: CN3 - CONNECTOR FOR SERIAL LINE AND POWER SUPPLY ........ 16
FIGURE 11: SERIAL COMMUNICATION DIAGRAM ............................................ 17
FIGURE 12: RS 232 POINT TO POINT CONNECTION EXAMPLE ...................... 17
FIGURE 13: RS 422 POINT TO POINT CONNECTION EXAMPLE ...................... 18
FIGURE 14: RS 485 POINT TO POINT CONNECTION EXAMPLE ...................... 18
FIGURE 15: RS 485 NETWORK CONNECTION EXAMPLE ..................................... 19
FIGURE 16: 4 WIRES CURRENT LOOP POINT TO POINT CONNECTION EXAMPLE 20
FIGURE 17: 2 WIRES CURRENT LOOP POINT TO POINT CONNECTION EXAMPLE 20
FIGURE 18: PASSIVE CURRENT LOOP NETWORK CONNECTION EXAMPLE ....... 21
FIGURE 19: TTL SERIAL COMMUNICATION AND POWER SUPPLY CONNECTION EXAMPLE 22
FIGURE 20: GPC® R94 CARD PHOTO ................................................................. 23
FIGURE 21: CN8 - CONNECTOR FOR OPTOCOUPLED INPUTS ....................... 24
FIGURE 22: DIGITAL INPUTS CONNECTION DIAGRAM ................................. 25
FIGURE 23: CN8 - GPC® R94 RELAY OUTPUTS CONNECTOR .......................... 26
FIGURE 24: RELAY OUTPUTS DIAGRAM ......................................................... 27
FIGURE 25: CN8 - GPC® T94 TRANSISTOR OUTPUTS CONNECTOR ............... 28
FIGURE 26: TRANSISTOR OUTPUTS DIAGRAM ............................................... 29
FIGURE 27: CN6 - A/D CONVERTER INPUT CONNECTOR ... ........................... 30
FIGURE 28: A/D CONVERTER INPUT DIAGRAM ............................................. 31
FIGURE 29: LEDS TABLE ................................................................................. 32
FIGURE 30: GPC® T94 CARD PHOTO ............................................................... 33
FIGURE 31: JUMPERS SUMMARIZING TABLE ................................................... 36
FIGURE 32: 3 AND 5 PINS JUMPERS TABLE ................................................... 36
FIGURE 33: GPC® R94 JUMPERS LOCATION ................................................... 37
FIGURE 34: 2 PINS JUMPERS TABLE ............................................................... 38
FIGURE 35: MEMORY SELECTION TABLE ....................................................... 38
FIGURE 36: GPC® T94 JUMPERS LOCATION ................................................... 39
FIGURE 37: SERIAL COMMUNICATION DRIVERS LOCATION ....................... 41
FIGURE 38: BAUD RATE SELECTION TABLE .................................................... 51
FIGURE 39: COMMUNICATION MODES SELECTION TABLE ......................... 51
FIGURE 40: NAME SETTING TABLE ................................................................. 51
FIGURE 41: TIMED SET COMMAND ............................................................... 58
FIGURE 42: TIMED CLEAR COMMAND ............................................................ 59
FIGURE 43: TIMED SQUARE WAVE COMMAND ............................................. 61
FIGURE 44: TIMED SQUARE WAVE COMMAND ............................................. 62
FIGURE 45: SQUARE WAVE COMMAND ....................................................... 63
FIGURE 46: MAXIMUM NUMBER OF MESSAGES MEMORIZABLE IN EEPROM 65
FIGURE 47: RTC INITIALIZATION BYTES VALIDITY RANGE ............................................................... 67
FIGURE 48: CONNECTION EXAMPLE WITH TWO 1-WIRE® DEVICES ................................................... 69
FIGURE 49: POSSIBLE CONNECTIONS DIAGRAM ................................................................................ 77
FIGURE A1: ALB x94 COMMANDS SUMMARIZING TABLE 1 ............................................................... A-1
FIGURE A2: ALB x94 COMMANDS SUMMARIZING TABLE 2 ............................................................... A-2
INTRODUCTION

The use of these devices has turned - IN EXCLUSIVE WAY - to specialized personnel.

The purpose of this handbook is to give the necessary information to the cognizant and sure use of the products. They are the result of a continual and systematic elaboration of data and technical tests saved and validated from the manufacturer, related to the inside modes of certainty and quality of the information.

The reported data are destined- IN EXCLUSIVE WAY - to specialized users, that can interact with the devices in safety conditions for the persons, for the machine and for the enviroment, impersonating an elementary diagnostic of breakdowns and of malfunction conditions by performing simple functional verify operations , in the height respect of the actual safety and health norms.

The informations for the installation, the assemblage, the dismantlement, the handling, the adjustment, the reparation and the contingent accessories, devices etc. installation are destined - and then executable - always and in exclusive way from specialized warned and educated personnel, or directly from the TECHNICAL AUTHORIZED ASSISTANCE, in the height respect of the manufacturer recommendations and the actual safety and health norms.

The devices can't be used outside a box. The user must always insert the cards in a container that respect the actual safety normative. The protection of this container is not threshold to the only atmospheric agents, but specially to mechanic, electric, magnetic, etc. ones.

To be on good terms with the products, is necessary guarantee legibility and conservation of the manual, also for future references. In case of deterioration or more easily for technical updates, consult the AUTHORIZED TECHNICAL ASSISTANCE directly.

To prevent problems during card utilization, it is a good practice to read carefully all the informations of this manual. After this reading, the user can use the general index and the alphabetical index, respectly at the begining and at the end of the manual, to find information in a faster and more easy way.

CARD VERSION

The present handbook is reported to the following versions:

- Board GPC® R94: version 221199 and later.
- Board GPC® T94: version 221199 and later.
- Firmware ALB x94: version 1.3 and later.

The validity of the bring informations is subordinate to the number of the card release. The user must always verify the correct correspondence among the two denotations. On the card the release number is present in more points both board printed diagram (serigraph) and printed circuit (for example near battery BT1 on solder side). The firmware version is labelled on the microprocessor, or it can be requested by the proper serial command.
GENERAL INFORMATION

The GPC® R94 or T94 belongs to the CPUs M Serie 22.5x82x90 mm size, and it is a powerful control Low-Cost module capable of operation in stand-alone mode or as an intelligent peripheral, and/or remoted, in a wider telecontrol or acquisition network. The GPC® R94 or T94 module is secured in a plastic mount for connection to Omega rails DIN 46277-1 and DIN 46277-3, thereby dispensing with the need of rack and allowing a less costly mounting direct to the electrical control panel. The facility in the performance can be carried on thanks to a wide range of software development tools with evolved language which, in an efficient and friendly environment, allow to work at the best using a standard PC. For example with BASCOM 8051 and SIM2051 is possible generate the applications in really fast time and with minimal investments.

The GPC® R94 or T94 is available with telecontrol firmware already programmed on the board. In this case through ALB (Abaco® Link BUS) it can be used as intelligent module in stand-alone mode or in network mode with a normal serial line. It is so possible, to realize applications with few I/O driven by a remote PC.

- Intelligent ABACO® BLOCK module, M serie.
- 22.5 x 82 x 90 mm size.
- Plastic mount for connection to DIN 46277-1 and DIN 46277-3 Ω rails.
- 14 MHz 89C2051 or 89C4051 CPU.
- 2K bytes or 2K bytes FLASH and 128 bytes SRAM.
- Code compatible with 51 family µP.
- Real Time Clock with 256 bytes of internal SRAM.
- Back Up circuitry for SRAM and RTC through LITHIUM battery.
- 1K bytes serial EEPROM.
- 9 Optocoupled Input TTL lines.
- Optocoupled external INT line.
- 4 outputs: 5A Relay in the R version.
  Optocoupled 4A, 45 Vdc NPN Darlington without heat sink and with back EMF protection diode, in version T.
- LEDs to visualize I/O lines status.
- All I/O connections are available on quick release connectors.
- 16 bits Timer Counter registers.
- 1 line 11 bits A/D Converter with analog input range = 0÷10Vdc and conversion time = 60 ms.
- 1 Serial line in TTL, RS 232, RS 422, RS 485 or Current Loop.
- Facility of operation in Idle or Power down mode.
- High efficiency on board switching power supply.
- +24 Vdc for optocoupled power supply.
- 5 or 5÷40 Vdc or 10÷24 Vac for on board logic.
- Protection against voltage peaks by TransZorb™.
- Wide range of base software and Development Tools enables the use of the card connected to any PC, making a special development system unnecessary. Among the available programs there are: “C” Compiler, Assembler, BASIC Compiler BASCOM LT, HTC-51, etc.
FIGURE 1: GPC® R 94 BLOCK DIAGRAM

CPU
AT89c2051 or AT89c4051

DRIVERS
TTL
RS232
RS422 - RS485
CURRENT LOOP

8 INPUT LINES DRIVER

OPTO-COUPLERS

CN3
SERIAL LINE

CN6
1 A/D LINE or 2 TTL I/O LINES

IC11
EEPROM

IC9
SRAM+RTC

SWITCHING POWER SUPPLY

4 OUTPUT LINES DRIVER

RELAYS

CN7
9 INPUT LINES

CN8
4 OUTPUT LINES

ON BOARD BATTERY

+5 Vdc

+V opto

I²C bus

GPC® R/T 94 Rel. 5.40
Here follows a description of the board's sections and the operations they perform. To easily locate such section on verify their connections please refer to figures 1 and 2.

**CPU**

The **GPC® R/T94** can use all version of ATMEL microprocessors namely 89c2051 or 89c4051. These 8 bit microprocessors are provided with an internal FLASH EPROM of respectively 2K Bytes or 4K Bytes and are code compatible with the 8051 INTEL and so they have an extended instruction set, fast execution and data manipulation time, easy use of all kind of memory, several hardware peripherals inside and an efficient interrupt management. The most important features of the described microprocessors, are:

- μP AT89c2051 AT89c4051:
  - 8 bit CPU;
  - 2K or 4K bytes of ROM FLASH;
  - 128 bytes of internal SRAM;
  - 15 programmable I/O lines;
  - 2 x 16 bits Timer/Counters;
  - 5 interrupts sources;
  - 2 priority level for interrupts;
  - 1 full duplex serial line;
  - 1 analog comparator;
  - Low Power Idle and Power Down Modes;

The **GPC® R/T94** by default is delivered with an AT89c4051 already programmed with telecontrol program called **ALB x94**. In this case, thanks to ALB (**ABACO®** Link BUS) communication protocol, it can be used like an intelligent peripheral, alone or in net by a standard serial line; so even applications with limited number of I/O, driven by a normal PC, could be realized in an easy way. For further information, please refer to specific documentation of the manufacturing company or to appendix B of this manual.

**POWER SUPPLY SECTION**

One of the most important features of **GPC® R/T94** is its on board power supply circuitry; the card can be powered in two different ways: +5 Vdc (without power supply section) or 10÷24Vac (switching section). The power supply circuit generates the necessary voltages for the card, starting from all the standard industrial source like mains, power transformer, battery, solar cell, etc. The User must provide also a 24 Vdc voltage for the optocoupled section. The power supply type must be specified at the moment of the order. The power supply circuit was designed for reducing the consumption (the microprocessor power down and idle mode is available) and for increasing the electrical noise immunity. Remember that on board there is a protection circuit against voltage peaks by **TransZorb™**.
**Figure 2: GPC® T 94 Block Diagram**

- **CPU**
  - AT89c2051 or AT89c4051
  - Connections:
    - 8 input lines driver
    - 1 A/D line or 2 TTL I/O lines

- **Drivers**
  - TTL
  - RS232
  - RS422 - RS485
  - Current loop

- **IC9**
  - SRAM + RTC
  - Drivers:
    - TTL
    - RS232
    - RS422 - RS485

- **IC11**
  - EEPROM
  - +5 Vdc

- **Switching Power Supply**
  - +V opto

- **CN3**
  - Serial line

- **CN6**
  - 1 A/D line or 2 TTL I/O lines

- **CN4**
  - Power

- **CN7**
  - 9 input lines

- **CN8**
  - 4 output lines

- **Opto-couplers**
  - Connections:
    - ON board battery
CLOCK

Installed on GPC® R/T 94 boards there are two separated circuitries, based on two different quartzes, to generate clock signals for the microprocessor (14.7456 MHz) and for the Real Time Clock (32.768 KHz). The choice of using two circuits and two separated crystals, has the advantage to change the microprocessor working speed (when best performances are required) without additional changes in software, firmware, etc.

MEMORY DEVICES

The total memory amount that the card can support can be up to 1280 Bytes. Normally the card is equipped with 512 bytes of serial EEPROM and 256 bytes of serial SRAM+RTC. With the on board back up circuit there is the possibility to keep the 256 SRAM+RTC data, also when power supply is failed; in this way the card is always able to maintain parameters, logged data, system status and configuration, etc. without using expensive external UPS. The back up circuit is based on an internal LITHIUM battery.

- **Real Time Clock:** The IC 9 SRAM module, is provided with on board lithium battery and with Real Time Clock which manages time (hours, minutes, seconds) and date (day, month, year, day of the week).

- **Serial EEPROM:** With the IC11 EEPROM module (range 512÷1024 bytes), there is the possibility to keep data also when power supply failed; in this way the card is always able to maintain parameters, logged data, system status and configuration, etc. without using expensive external UPS. This component has a default size of 512 bytes.

For further information about peripheral device please refer to the technical documentation of the manufacturing company.

GPC® R94 OUTPUT SECTION

This section has 4 output lines, available on a comfortable quick release connector, driven by PNP transistor. These components are managed through 4 CPU pins, following the informations explained in the "HARDWARE DESCRIPTION" and "SOFTWARE DESCRIPTION" chapters. Each line has an own LED to show the line status and it drives a 5A N.O. Relay.

GPC® T94 OUTPUT SECTION

This section has 4 output lines, available on a comfortable quick release connector, driven by optocoupler. These components are managed through 4 CPU pins, following the informations explained in the "HARDWARE DESCRIPTION" and "SOFTWARE DESCRIPTION" chapters. Each line is galvanically isolated and has an own LED to show the line status and it can drive a 4A, 45Vdc Darlington transistor that is connected in Open Collector with recovery rectifier.
INPUT SECTION

Throught a 74HCT166 (driven by 3 CPU pins) and one CPU pin the User can read the 9 input lines of the card. Each NPN input lines is optcoupled and its state is shown by a proper LED. The optocoupled section is powered by the +Vopto voltage that is provided by the power supply section. Please remaek that if necessary it is possible to acquire the signals of three lines very quickly bypassing the serializer and connecting opportunaly the desired lines. For further informations please contact directly the grifo® technical personnel.

FIRMWARE ALB X94

GPC® R/T 94 boards are delivered by default with the firmware ALB x94 (ABACO® Link BUS), that allows to manage all the on-board resources through several commands sent to the serial line. A remarkable feature is the implementation of two separated 1-Wire® protocol communication buses made using the I/O TTL lines. It is so possible to manage two different 1-Wire® devices through the serial line of GPC® R/T 94 (temperature sensors, memories, Dallas iButton™, etc.).

It is also supported a Master-Slave communication mode that allows to remote control single modules from great distance, realizing a telecontrol network managed by an unique master unit (PC, PLC, GPC® serie board, etc.).

A/D CONVERTER AND TTL I/O LINES

GPC® R/T 94 boards are provided with an analog input for A/D conversion, based on a circuitry that, by means of the microprocessor inside analog comparator, can acquire a signal in the range 0÷10 V, 0÷20 mA or 4÷20 mA, with 11 bits of resolution and a maximum conversion time of 60 msec. This analog input is reachable on a comfortable quick release connector, to interface easily with the external world.

In addition, it is possible to bypass the analog circuitry by moving some jumpers, and connect the microprocessor's pins directly to the connector, obtaining two I/O TTL lines for general purpose or just a direct access to the inputs of the comparator.

SERIAL COMMUNICATION SELECTION

An asyncronous serial line is available on GPC® R/T94 and it can be buffered in TTL, RS 232, RS 422, RS 485 or Current Loop. By hardware can be selected which one of these electric standards is used, by software the serial line can be programmed to operate with all the standard physical protocols, in fact the bits per character, parity, stop bits and baud rates can be decided by setting opportunues CPU internal registers. For further informations please refer to the manufacturer documentation or to appendix B of this manual.
TECHNICAL FEATURES OF GPC® R94

GENERAL FEATURES OF GPC® R94

On board resources:
- 9 NPN optocoupled digital inputs
- 4 N.A. 5 A relays digital outputs
- 2 TTL input/output
- 1 NPN external optocoupled interrupt
- 1 analog comparator
- 1 A/D Converter signal
- 1 Real Time Clock
- 1 serial line TTL, RS 232, RS 422-485 or Current Loop

Memory devices:
- IC 9: serial RTC+SRAM 256 bytes
- IC 11: serial EEPROM from 512 bytes to 1024 bytes

CPU:
- Atmel AT89C2051 or AT89C4051

Clock frequency:
- 14.7456 MHz (CPU)
- 32.768 KHz (Real Time Clock)

A/D resolution: 11 bits

A/D conversion time: Max 60 ms

PHYSICAL FEATURES OF GPC® R94

Size (W x H x D):
- 111 x 90 x 22.5 mm
- 111 x 102 x 22.5 mm (included container for Ω rails)

Mounting:
- On Ω rails type DIN 46277-1 and DIN 46277-3

Weight:
- 182 g (basic version)

Connectors:
- CN3: 4+4 pins AMP Mod II male 90°
- CN4: 4 pins quick release 90°
- CN6: 2 pins quick release 90°
- CN7: 10 pins quick release 90°
- CN8: 6 pins quick release 90°

Female connector for CN3: grifo® code: CKS.AMP8 (kit made of a female AMP Mod II 4+4 8 ways connector and pins to crimp)

Temperature range:
- from 0 to 50 Centigrad degrees

Relative humidity:
- 20% up to 90% (without condense)
Figure 3: Components map GPC® R94 (component side)

Figure 4: Components map GPC® R94 (solder side)
ELECTRIC FEATURES OF GPC® R94

**Version without power supply**

**Supply voltage:**
- +5 Vdc  (control logic)
- +24 Vdc  (+V opto)

**Version with switching power supply**

**Input supply voltage:**
- 10÷40 Vdc or 8÷24 Vac  (control logic)
- 24 Vdc  (+V opto)

**Output supply voltage:**
- +5 Vdc; 740 mA
- +Vopto 100 mA

**Current consumption:**
- 260 mA max  (+5 Vdc)
- 40 mA max  (+V opto)

**Maximum current on relays:**  5A

**Maximum voltage on relays:**  30 Vdc / 250 Vac

**NPN inputs minimum current:**  2 mA

**Back Up current:**  3.5 µA

**Back Up on board battery:**  3 V; 180 mAh; mod. CR 2032

**Analog input:**
- 0÷10 V; 0÷20 mA; 4÷20 mA

**Analog input impedance:**  1 MΩ

**RS422-485 termination network:**
- Line termination resistor  =  120 Ω
- Positive pull up resistor  =  3.3 KΩ
- Negative pull down resistor  =  3.3 KΩ
TECHNICAL FEATURES OF GPC® T94

GENERAL FEATURES OF GPC® T94

On board resources: 9 NPN optocoupled digital inputs
4 open collector darlington NPN digital outputs, with back EMF protection diode
2 TTL input/output
1 NPN external optocoupled interrupt
1 analog comparator
1 A/D Converter signal
1 Real Time Clock
1 serial line TTL, RS 232, RS 422-485 or Current Loop

Memory devices:
IC 9: serial RTC+SRAM 256 bytes
IC 11: serial EEPROM from 512 bytes to 1024 bytes

CPU: Atmel AT89C2051 or AT89C4051

Clock frequency: 14.7456 MHz (CPU)
32.768 KHz (Real Time Clock)

A/D resolution: 11 bits

A/D conversion time: Max 60 ms

PHYSICAL FEATURES OF GPC® T94

Size (W x H x D): 111 x 90 x 22.5 mm
111 x 102 x 22.5 mm (included container for Ω rails)

Mounting: On Ω rails type DIN 46277-1 and DIN 46277-3

Weight: 171 g (basic version)

Connectors:
CN3: 4+4 pins AMP Mod II male 90°
CN4: 4 pins quick release 90°
CN6: 2 pins quick release 90°
CN7: 10 pins quick release 90°
CN8: 6 pins quick release 90°

Female connector for CN3: grifo® code: CKS.AMP8 (kit made of a female AMP Mod II 4+4 8 ways connector and pins to crimp)

Temperature range: from 0 to 50 Centigrad degrees

Relative humidity: 20% up to 90% (without condense)
ELECTRIC FEATURES OF GPC® T94

Version without power supply
Supply voltage: +5 Vdc (control logic)
+24 Vdc (+V opto)

Version with switching power supply
Input supply voltage: 10÷40 Vdc o 8÷24 Vac (control logic)
24 Vdc (+V opto)
Output supply voltage: +5 Vdc; 930 mA
+Vopto 100 mA
Current consumption: 70 mA max (+5 Vdc)
40 mA max (+V opto)
Maximum current on Transistor: 4A not continuative (*)
Maximum voltage on Transistor: 45 Vdc (*)
Maximum power on Transistor: 1.25 W (*)
NPN inputs minimum current: 2 mA
Back Up current: 3.5 µA
Back Up on board battery: 3 V; 180 mAh; mod. CR 2032
Analog input: 0÷10 V; 0÷20 mA; 4÷20 mA
Analog input impedance: 1 MΩ
RS422-485 termination network: Line termination resistor = 120 Ω
Positive pull up resistor = 3.3 KΩ
Negative pull down resistor = 3.3 KΩ

(*) These values are refferred to a 20 ºC work temperature
Figure 5: Components map GPC® T94 (component side)

Figure 6: Components map GPC® T94 (solder side)
INSTALLATION

In this chapter there are the information for a right installation and correct use of the card. The User can find the location and functions of each connector, jumper and some explanatory diagrams.

CONNECTIONS

The **GPC® R/T94** module has 5 connectors that can be linked to other devices or directly to the field, according to system requirements. In this paragraph there are connectors pin out, a short signals description (including the signals direction) and connectors location. To easily locate them please refer to figures 8 and 9, while for further informations about the connections please refer to the following figures.

**CN4 - POWER SUPPLY CONNECTOR**

CN4 is a 4 pins, quick release, screw terminal connector. CN4 is used to power the card when on **GPC® R94** or **GPC® T94** there is the switching power supply section. If switching power supply section is not present the card must be powered through CN3 connector.

![Diagram of CN4 connector](image)

**Figure 7: CN4 - Power Supply Connector**

Signals description:

<table>
<thead>
<tr>
<th>Signal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+V opto</td>
<td>+V opto power supply.</td>
</tr>
<tr>
<td>GND opto</td>
<td>Opto ground signal.</td>
</tr>
<tr>
<td>Vac</td>
<td>Low voltage power supply line +10÷40 Vdc or 10÷24Vac.</td>
</tr>
<tr>
<td>Vac</td>
<td>Low voltage power supply line +10÷40 Vdc or 10÷24Vac.</td>
</tr>
</tbody>
</table>
**Figure 8:** LEDs, Connectors, etc. Location on GPC® R94

**Figure 9:** LEDs, Connectors, etc. Location on GPC® T94
**CN3 - CONNECTOR FOR SERIAL LINE AND POWER SUPPLY**

If the GPC® R94 or GPC® T94 has the switching power section mounted, through CN3 the user can draw the two galvanically isolated voltages and can use them for external loads supply.

In the other case (switching not present) through CN3 the user must give the two voltages (+5 Vdc and +Vopto) for card power supply. Also, on CN3 connector are available the buffered signals for TTL, RS 232, RS 422, RS 485 or Current Loop serial line communication, each of the listed electric protocols (CCITT normatives compliant). All the signals are placed in order to reduce interference and electrical noise. The female connector can be ordered directly to grifo® (code CKS.AMP8) or to AMP (female AMP 4+4 pins connector Mod II P/N: 280365; pins to be crimped as loose pieces P/N:182206-2).

![Diagram of CN3 connector](image)

### Signals description:

RX TTL = I - Receive Data of TTL serial line.

TX TTL = O - Transmit Data of TTL serial line.

RXD RS 232 = I - Receive Data of RS 232 serial line.

TXD RS 232 = O - Transmit Data of RS 232 serial line.

RX- RS 422 = I - Receive Data Negative of RS 422 serial line.

RX+ RS 422 = I - Receive Data Positive of RS 422 serial line.

TX- RS 422 = O - Transmit Data Negative of RS 422 serial line.

TX+ RS 422 = O - Transmit Data Positive of RS 422 serial line.

RX- RS 485 = I/O - Transmit Receive Data Negative of RS 485 serial line.

TXRX+ RS 485 = I/O - Transmit Receive Data Positive of RS 485 serial line.

RX- C.L. = I - Receive Data Negative of Current Loop serial line.

RX+ C.L. = I - Receive Data Positive of Current Loop serial line.

TX- C.L. = O - Transmit Data Negative of Current Loop serial line.

TX+ C.L. = O - Transmit Data Positive of Current Loop serial line.

+5 Vdc = I/O - Positive terminal of +5 Vdc power supply.

GND = - Digital ground signal.

+V opto = I/O - Positive terminal of +V opto power supply.

GND opto = - Ground signal for +V opto power supply.
**FIGURE 11: SERIAL COMMUNICATION DIAGRAM**

**FIGURE 12: RS 232 POINT TO POINT CONNECTION EXAMPLE**
NOTE

In RS 485, using firmware ALB x94, only 9 bit Master-Slave communication mode is possible.
Please remark that in a RS 485 network two forcing resistors must be connected across the net and two termination resistors (120 Ω) must be placed at its extrem, respecveely near the Master unit and the Slave unit at the greatest distance from the Master.

Forcing and terminating circuitry is installed on GPC® R/T943 board. It can be enabled or disabled through specific jumers, as explained later.

For further informations please ref to TEXAS INSTRUMENTS Data-Book, "RS 422 and RS 485 Interface Circuits", the introduction about RS 422-485.
FIGURE 16: 4 WIRES CURRENT LOOP POINT TO POINT CONNECTION EXAMPLE

FIGURE 17: 2 WIRES CURRENT LOOP POINT TO POINT CONNECTION EXAMPLE

NOTE

In 2 wires Current Loop, using firmware ALB x94, only 9 bit Master-Slave communication mode
Possible Current Loop connections are two: 2 wires and 4 wires. These connections are shown in figures 16 and 17 where it is possible to see the voltage for VCL and the resistances for current limitation (R). The supply voltage varies in compliance with the number of connected devices and voltage drop on the connection cable.

The choice of the values for these components must be done considering that:
- circulation of a 20 mA current must be guaranteed;
- potential drop on each transmitter is about 2.35 V with a 20 mA current;
- potential drop on each receiver is about 2.52 V with a 20 mA current;
- in case of shortcircuit each transmitter must dissipate at most 125 mW;
- in case of shortcircuit each receiver must dissipate at most 90 mW.

For further info please refer to HEWLETT-PACKARD Data Book, (HCPL 4100 and 4200 devices).
**Figure 19:** TTL SERIAL COMMUNICATION AND POWER SUPPLY CONNECTION EXAMPLE
FIGURE 20: GPC® R94 CARD PHOTO
CN7 - CONNECTOR FOR OPTOCOUPLED INPUTS

CN7 is a 10 pins, quick release, screw terminal connector. CN7 is used to connect the 9 optocoupled NPN input signals that the card manages. 8 inputs are connected to the 74HCT166 while the AUX input is connected to the CPU. On the connector are available the open collector inputs and the opto ground signal.

![Diagram of CN7 connector]

**Figure 21: CN8 - Connector for optocoupled inputs**

Signals description:

<table>
<thead>
<tr>
<th>IN0÷IN7</th>
<th>AUX</th>
<th>GND opto</th>
</tr>
</thead>
<tbody>
<tr>
<td>I - Open collector NPN input, connected to 74HCT166 (range H÷A).</td>
<td>I - Open collector NPN input, connected to CPU P3.2.</td>
<td>- Opto ground signal.</td>
</tr>
</tbody>
</table>
The inputs available on the board are optocoupled, each one driving a LED for visual feed-back (the LED will light when the input will be connected to the ground signal). The inputs are suitable for NPN type drivers, to connect them to PNP drivers you should interpose a PBI 01 module. The 9 inputs section is shown in the diagram below. Optocouplers power supply may be granted by connector CN4.

**Figure 22: Digital Inputs Connection Diagram**
**CN8 - GPC® R94 RELAY OUTPUTS CONNECTOR**

CN8 is a 6 pins, quick release, screw terminal connector. On CN8 are available the N.O. 4 relays outputs and their relative commons. The maximum external load for each line is 5 A with a maximum tension of 30 Vdc or 250 Vac. The outputs are managed through a set of microprocessor I/O pins, opportunaly buffered, carefully selected to simplify the software management (for further informations please refer to the chapter “PERIPHERAL DEVICES SOFTWARE DESCRIPTION”)

![CN8 - GPC® R94 RELAY OUTPUTS CONNECTOR](image)

**FIGURA 23: CN8 - GPC® R94 RELAY OUTPUTS CONNECTOR**

*Legenda:*

<table>
<thead>
<tr>
<th>OUT n (Px.y)</th>
<th>COMMON n, m</th>
</tr>
</thead>
<tbody>
<tr>
<td>O - N.O. contact of relay n, driven by the pin Px.y of the microcontroller;</td>
<td>- Common contact of relays n and m.</td>
</tr>
</tbody>
</table>
Each relay output signal is connected to a LED whose purpose is visual feedback (the LED will light whenever the relay contact is closed). The relays are driven by 4 PNP transistors that are managed through microcontroller's I/O pins.

**FIGURA 24: RELAY OUTPUTS DIAGRAM**
CN8 - GPC® T94 TRANSISTOR OUTPUTS CONNECTOR

CN8 is a 6 pins, quick release, screw terminal connector. On CN8 are available the 4 open collector Darlington NPN transistor outputs and their relative commons (emitter). The maximum external load for each line is 4 A not continuative with 45 Vdc. The transistors, being without heat sink, can drive in continuative way a resistive load that absorbs at most 600 mA with a maximum voltage if 45 Vdc only if the work temperature is 20°C. All the signals are provided with a back EMF protection diode which suppresses eventual inductive voltages when loads like power relays, solenoids etc. are driven. In this case the load supply must be connected to the +VL signal. The outputs are managed through a set of microprocessor I/O pins, opportunaly buffered, carefully selected to simplify the software management (for further informations please refer to the chapter “PERIPHERAL DEVICES SOFTWARE DESCRIPTION”.

Signals description:

OUT n (Px.y) = O - Open collector contact of n-th NPN transistor, driven by microcontroller’s pin Px.y.
COMMON = - Common emitter of transistors.
+VL = I - Back EMF protection diodes power supply (power supply of load).
Each output is optocoupled and it has its own LED that shows the line status (the LED is turned on when the transistor is in conduction).

The final stage of such outputs is made of an NPN Darlington transistor in Open Collector, provided with back EMF protection diode, all the output's emitters are connected to an unique common point.

**Figura 26: Transistor outputs diagram**
CN6 - A/D CONVERTER INPUT CONNECTOR AND TTL I/O LINES

CN6 is a 2 pins, quick release, screw terminal connector used to connect the two multi-purpose signals (A/D Converter input, I/O TTL or analog comparator inputs) of GPC® R/T94.

The on-board circuitry is configurable through jumpers J5 and J6, as explained later, to solve the User problematics without further expenses.

If the A/D conversion is selected, please remark that a filtering capacitor is connected and that the input signal range must be 0÷10 V or 0÷20 mA or 4÷20 mA, while the conversion is managed through the microprocessor's analog comparator registers, as described in the chapter “PERIPHERAL DEVICES SOFTWARE DESCRIPTION”.

**FIGURE 27: CN6 - A/D CONVERTER INPUT CONNECTOR**

**Signals description:**

Px.y (ANy) = I/O - I/O TTL signal connected to microcontroller's pin Px.y or input ANy of the analog comparator.


AGND = - Analog ground signal.
**Figura 28: A/D converter input diagram**

- **CPU**
  - +5 Vdc
  - Constant current generator

- **Analog input divider**
  - J5, J6
  - 1 A/D line

- **Integrator**
  - AN0 / P1.0
  - AN1 / P1.1

- **R5**
  - 12
  - 13

- **CN6**
  - PIN 1
  - PIN 2
  - 2 TTL I/O lines
LEDS

On GPC® R/T 94 boards there are 15 LEDs that shows the card status, as described in the following table:

<table>
<thead>
<tr>
<th>LEDs</th>
<th>Colour</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1÷LD4</td>
<td>Red</td>
<td>They indicate the status of digital outputs 0÷3. The LED is on when the output OUTn is connected to COMMON n.</td>
</tr>
<tr>
<td>LD5÷LD12</td>
<td>Green</td>
<td>They indicate the status of digital input 0÷7. The LED is on when the input INPn is connected to GND opto.</td>
</tr>
<tr>
<td>LD13</td>
<td>Green</td>
<td>It indicates the status of AUX signal. The LED is on when the input AUX is connected to GND opto.</td>
</tr>
</tbody>
</table>

**Figure 29: LEDs table**

LEDs LD1÷LD12 are visible on one edge of the card, while LED LD13 is visible only removing the cover of the container.
The main purpose of LEDs is to inform the User about card status, with a simple visual indication and in addition to this, LEDs make easier the debug and test operations of the complete system. To recognize the LED location on the card, please refer to figures 8 and 9.

INTERRUPTS

A remarkable feature of GPC® R/T 94 cards is the powerful interrupt management. Here follows a short description of which devices can generate interrupts and their modalities; for further informations about interrupts management please refer to the microprocessor data sheet or to the appendix B of this manual.

- AUX input -> Generates an interrupt on microcontroller's pin /INT0 (P3.2).
- CPU peripherals -> Generate an internal interrupt. In particular the possible internal interrupt sources are the sections: Timer/Counter and serial line.

An interrupt manager that allows to activate, deactivate and mask the interrupt sources is installed on the board. It can also manage contemporary interrupts. This way the User can always react promptly and efficiently to any external event, being also able to decide a priority for the several interrupt sources.

BACK UP

On GPC® R/T 94 is mounted a Lithium battery that keeps data on SRAM+RTC also when power supply fails. With jumper J7 the User can connect or disconnect the Back Up whenever it is needed, so the battery life time is increased. To easily locate the battery please refer to figures 8 and 9.
Figure 30: GPC® T94 card photo
I/O CONNECTION

To prevent possible connecting problems between GPC® R/T 94 and the external systems, the User has to read carefully the previous paragraph information and he must follow these instructions:

- For RS 232, RS 422, RS 485 and Current Loop signals the User must follow the standard rules of each one of these protocols;

- For all TTL signals the User must follow the rules of this electric standard. The connected digital signal must be always referred to card digital ground and if an electric insulation is necessary, then an opto coupled interface must be connected. For TTL signals, the 0V level corresponds to logic state 0, while 5V level corrisponds to logic state 1.

- The analog input (A/D Converter section) must be connected to signals within the ranges: 0÷10 V, 0÷20 mA o 4÷20mA according to card configuration. Please remark that such input, available on connector CN6, is provided with a low width filtering capacitor that warrants a greater stability of the signal acquired.

- To the optocoupled input signals only the contacts to be acquired must be connected. Such contacts (relays, switches; etc.) must connect or must not connect the input signal INx to GND opto. For logic signals correspondance, the open contact corresponds to logic state 1, while the closed contact corresponds to logic state 0. Correspondance is compliant to the NPN normative.

- The relays output signals, available only on GPC® R94, must be connected directly to the load to be driven (power relays, etc.). The boards provides the normally open contact, for which maximum current is 5A with a tension that can be as high as 30 Vdc or 250 Vac.

  To give the chance to drive different loads with different supplies, two different COMMON contacts are available, that connect two different groups of relays.

- The Darlington NPN output signals, available only on GPC® T94, must be connected directly to the load to be driven (power relays, etc.). The boards provides the Open Collector output line, for which maximum current is 4A not continuative with a tension that can be as high as +45 Vdc.

  The transistors, being without heat sink, can drive in continuative way a resistive load that absorbs at most 600 mA with a maximum voltage as high as 45 Vdc only if the work temperature is 20° C.
ANALOG INPUTS SELECTION

One of the GPC® R/T 94 particular features is the possibility to acquire tension and/or current signals for the A/D inpus. The CURRENT signals selection is obtained through proper resistor mounted on R5 position (option .8420).
If the resistor is not mounted (default) the channel can acquire a TENSION signal in the range 0÷10 Vdc (default), instead if the resistor is mounted the channel can acquire a CURRENT signal.
The resistors value for the CURRENT/VOLTAGE converter section is calculated with the following formula:

\[ R = \frac{10 \text{ V}}{I_{\text{max}}} \]

Normaly the precision CURRENT/VOLTAGE resistor value is 499Ω suitable for 0÷20 mA analog inputs. For eventual requirements outside these standard ranges please contact grifo®.
Please refer to figures 8 and 9 for the resistor location.

POWER SUPPLY SECTION

One of the most important features of GPC® R/T94 is its on board power supply circuitry; the card can be powered in two different ways to let the User solve easier the problem to supply the board in any working condition:

*Without switching power supply (default):*

+V opto: Supplies the optocouplers of the board's input section; it must be +24 Vdc and must be provided through pins 1 and 2 of CN4 or pins 2 and 8 of CN3.

+5 Vdc: Supplies the control logic and the board's output section: it must be +5 Vdc ±5% and must be provided through pins 1 and 7 of CN3.

*With switching power supply (option .SW):*

+V opto: Supplies the optocouplers of the board's input section; it must be +24 Vdc and must be provided through pins 1 and 2 of CN4 or pins 2 and 8 of CN3.

Vac: Supplies the control logic and the board's output section through the on board switching power supply: it must be 10÷40 Vdc or 8÷24 Vac (polarity is irrelevant) and must be provided through pins 3 and 4 of CN4. The power supply circuit generates the necessary voltages for the card, starting from all the standard industrial source like mains, power transformer, battery, solar cell, etc. It is possible to supply external loads with the +5 Vdc tension available on pins 1 and 7 of CN3.
Please remark that the on board switching supply section is provided with its own diode bridge, so if the supply is continuous tension the ground digital signal (GND) doesn't have the same potential of the ground signal available on CN4.

To warrant the maximum electrical noise immunity and so the correct board's functions, power supply voltages must be galvanically isolated; to assure this condition even starting from mains supply the power supply EXPS-2 can be ordered.
The power supply type must be specified at the moment of the order and can be installed only by grifo® technicians. The power supply circuit was designed for reducing the consumption (the microprocessor power down and idle mode is available) and for increasing the electrical noise immunity. On board there is a protection circuit against voltage peaks by TransZorb™.
JUMPERS

On GPC® R/T94 there are 8 jumpers, two of them are solder jumpers, for card configuration. Connecting these jumpers, the user can define for example the memory type and size, the peripheral devices functionality and so on. Below there is the jumpers list, location and function:

![Figure 31: Jumpers Summarizing Table](image)

The following tables describe all the right connections of GPC® R/T94 jumpers with their relative functions. To recognize these valid connections, please refer to the board printed diagram (serigraph) or to figures 3÷6 of this manual, where the pins numeration is listed; for recognizing jumpers location, please refer to figures 33 and 36. The "*" used in the following tables, denotes the default connection, or on the other hand the connection set up at the end of testing phase, that is the configuration the user receives.

3 AND 5 PINS JUMPERS

![Figure 32: 3 and 5 Pins Jumpers Table](image)

Connection in position 3-3 for jumpers J5 and J6 means to connect the pins number 3 of both the jumpers.
FIGURE 33: GPC® R94 JUMPERS LOCATION
2 PINS JUMPERS

<table>
<thead>
<tr>
<th>JUMPERS</th>
<th>CONNECTION</th>
<th>PURPOSE</th>
<th>DEF.</th>
</tr>
</thead>
<tbody>
<tr>
<td>J2, J3</td>
<td>not connected</td>
<td>They configure the serial line in RS232, RS422, RS485 or Current Loop (please see paragraph “SERIAL COMMUNICATION” for further informations).</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>They configure the serial line in TTL (please see paragraph “SERIAL COMMUNICATION” for further informations).</td>
<td></td>
</tr>
<tr>
<td>J7</td>
<td>not connected</td>
<td>Does not connect Lithium battery BT1 to the Back Up circuitry.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>Connects Lithium battery BT1 to the Back Up circuitry.</td>
<td></td>
</tr>
<tr>
<td>JS1, JS2</td>
<td>not connected</td>
<td>Do not connect the termination and forcing circuitry to the RS 422 reception line or RS 485 serial line.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>Connect the termination and forcing circuitry to the RS 422 reception line or RS 485 serial line.</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 34: 2 PINS JUMPERS table**

The "*" denotes the default connection, which is set up at the end of testing phase.

MEMORY SELECTION

Le **GPC® R/T94** can manage up to 5504 bytes of memory in several configurations. For the details of the possible configurations please refer to the following table:

<table>
<thead>
<tr>
<th>IC</th>
<th>DEVICE</th>
<th>SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>AT89C2051</td>
<td>2K Bytes (FLASH EPROM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128 bytes (SRAM)</td>
</tr>
<tr>
<td>5</td>
<td>AT89C4051</td>
<td>4K Bytes (FLASH EPROM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128 bytes (SRAM)</td>
</tr>
<tr>
<td>9</td>
<td>SRAM+RTC</td>
<td>256 Bytes</td>
</tr>
<tr>
<td>11</td>
<td>EEPROM</td>
<td>512÷1K Bytes</td>
</tr>
</tbody>
</table>

**Figure 35: MEMORY SELECTION table**

**GPC® R/T94** is delivered in its default configuration with microprocessor AT89C4051, SRAM+RTC device installed and 512 bytes of EEPROM; every different configuration can be performed by the User in autonomy (except for EEPROM IC11) or required in ordering phase. Below are reported the codes for the memory options available:

.**EE08** -> 1K serial EEPROM

For further informations and prices of the options please contact **grifo®**, while to easily locate memory devices please refer to figures 8 and 9.
FIGURE 36: GPC® T94 JUMPERS LOCATION
SERIAL COMMUNICATION SELECTION

GPC® R/T 94 serial line can be buffered as TTL, RS 232, RS 422, RS 485 Current Loop. By hardware can be selected which one of these electric standards is used, through jumpers connection and drivers installation. By software the serial line can be programmed to operate with all the standard physical protocols, in fact the bits per character, parity, stop bits and baud rates can be decided by setting opportunes CPU internal registers. In the following paragraphs there are all the informations on serial communication configurations.

Some devices needed for RS 422, RS 485 and Current Loop configurations are not mounted on the board in standard configuration; this is why each fist non-standard (non-RS 232) serial configuration must be always performed by grifo® technicians. This far the User can change in autonomy the configuration following the informations below:

- SERIAL LINE IN RS 232 (option .RS232)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>IC12 = driver MAX 202</th>
</tr>
</thead>
<tbody>
<tr>
<td>J2, J3</td>
<td>=</td>
<td>not connected</td>
</tr>
<tr>
<td>J4</td>
<td>=</td>
<td>don’t care</td>
</tr>
<tr>
<td>JS1, JS2</td>
<td>=</td>
<td>not connected</td>
</tr>
</tbody>
</table>

- SERIAL LINE B IN CURRENT LOOP (option .CLOOP)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>IC12 = no device</th>
</tr>
</thead>
<tbody>
<tr>
<td>J2, J3</td>
<td>=</td>
<td>not connected</td>
</tr>
<tr>
<td>J4</td>
<td>=</td>
<td>don’t care</td>
</tr>
<tr>
<td>JS1, JS2</td>
<td>=</td>
<td>not connected</td>
</tr>
</tbody>
</table>

Please remark that Current Loop serial interface is passive, so it must be connected an active Current Loop serial line, that is a line provided with its own power supply. Current Loop interface can be employed to make both point-to-point and multi-point connections through a 2-wires or a 4-wires connection.

- SERIAL LINE B IN RS 422 (option .RS 422)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>IC12 = no device</th>
</tr>
</thead>
<tbody>
<tr>
<td>J2, J3</td>
<td>=</td>
<td>not connected</td>
</tr>
<tr>
<td>J4</td>
<td>=</td>
<td>position 2-3</td>
</tr>
<tr>
<td>JS1, JS2</td>
<td>=</td>
<td>(*1)</td>
</tr>
</tbody>
</table>

Status of signal P3.7, which is software managed, allows to enable or disable the transmitter as follows:

<table>
<thead>
<tr>
<th>P3.7</th>
<th>Logic state</th>
<th>Transmitter Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>0</td>
<td>Enabled</td>
</tr>
<tr>
<td>High</td>
<td>1</td>
<td>Disabled</td>
</tr>
</tbody>
</table>

In point-to-point connections, signal P3.7 can be always kept low (transmitter always enabled), while in multi-point connections transmitter must be enabled only when a transmission is requested.
Serial line in RS 232
Serial line in Current Loop

Serial line in RS 422
Serial line in RS 485

**Figure 37: Serial communication drivers location**
- SERIAL LINE B IN RS 485 (option RS 485)

<table>
<thead>
<tr>
<th>J2, J3</th>
<th>J4</th>
<th>JS1, JS2</th>
<th>IC12</th>
<th>IC13</th>
<th>IC14</th>
<th>IC15</th>
<th>IC16</th>
</tr>
</thead>
<tbody>
<tr>
<td>not connected</td>
<td>position 1-2</td>
<td>(*1)</td>
<td>= no device</td>
<td>= no device</td>
<td>= no device</td>
<td>= driver MAX 483 or SN 75176</td>
<td>= no device</td>
</tr>
</tbody>
</table>

In this modality the signals to use are pins 5 and 6 of connector CN3, that become transmission or reception lines according to the status of signal P3.7, managed by software, as follows:

- P3.7 = low level = logic state 0 -> transmitter enabled
- P3.7 = high level = logic state 1 -> transmitter disabled

This kind of serial communication can be used for multi-point connections, in addition it is possible to listen to own transmission, so the User is allowed to verify the success of transmission. In fact, any conflict on the line can be recognized by testing the received character after each transmission.

(*1) If using the RS 422 or RS 485 serial line, it is possible to connect the terminating and forcing circuit on the line by using JS1 and JS2. This circuit must be always connected in case of point-to-point connections, while in case of multi-point connections it must be connected only in the farthest boards, that is on the edges of the communication line.

- SERIAL LINE IN TTL (default configuration)

<table>
<thead>
<tr>
<th>J2, J3</th>
<th>J4</th>
<th>JS1, JS2</th>
<th>IC12</th>
<th>IC13</th>
<th>IC14</th>
<th>IC15</th>
</tr>
</thead>
<tbody>
<tr>
<td>connected</td>
<td>indifferent</td>
<td>not connected</td>
<td>= no device</td>
<td>= no device</td>
<td>= no device</td>
<td>= no device</td>
</tr>
</tbody>
</table>

When a reset or a power on occur, signal PWM1 is kept to a logic level high, so in one of these two cases driver RS 485 is receiving or RS 422 driver is disabled, avoiding eventual conflicts in communication.

For further information about serial communication please refer to the examples of figures 12÷18 and to appendix B of this manual.
SOFTWARE

A wide selection of software development tools can be obtained, allowing use of the GPC® R/T 94 card as a system for its own development, both in assembler and in other high level languages; in this way the user can easily develop all the requested application programs in a very short time. Generally all software packages available for the 51 microprocessors family can be used.

MICRO/ASM-51: macro cross assembler available for MS-DOS operating system in "ABSOLUTE" and "RELOCATABLE" version. In this "RELOCATABLE" version is supplied with a linker and a library manager.

MICRO/C-51: integer cross compiler for source files in standard ANSI C, available also for MS-DOS operating system. It produces a source assembly file compatible with MICRO/ASM-51 or with Intel macro relocatable assembler MCS 51.

MICRO/SLD-51: source level debugger and simulator. It is executed on P.C. without any additional hardware and it allows loading of HEX and SYMBOLIC files, breakpoint setting, instruction execution in trace mode, registers and memory dump, etc.

HI-TECH C: cross compiler for C source program. It is a powerful software tool that includes editor, C compiler, assembler, optimizer, linker, library, and remote symbolic debugger, in one easy to use integrated development environment.

BASCOM LT: cross compiler for BASIC source program. It is a powerful software tool that includes editor, BASIC compiler and simulator included in an easy to use integrated development environment for Windows.

DDS MICRO C 51: low cost cross compiler for C source program. It is a powerful software tool that includes editor, C compiler (integer), assembler, optimizer, linker, library, and remote debugger, in one easy to use integrated development environment. There are also included the library sources and many utilities programs.
PERIPHERAL DEVICES SOFTWARE DESCRIPTION

In the previous paragraphs are described the external registers addresses, while in this one there is a specific description of registers meaning and function (please refer to I/O addresses table, for the registers names and addresses values). For a more detailed description of the devices, please refer to manufacturing company documentation; for microprocessor internal peripheral devices, not described in this paragraph, refer to appendix B. In the following paragraphs the D7:D0 and .0:7 indications denote the eight bits of the combination used in I/O operations.

SERIAL EEPROM

For software management of serial EEPROM module of IC 11, please refer to specific documentation or to demo programs supplied with the card. The first 32 (0:31) bytes of serial EEPROM are reserved for software tools use, so they can’t be neither read nor written by User program. The User must realize a serial communication with I2C bus standard protocol, through two I/O microprocessor pins. The only necessary information is the electric connection:

P1.3  <->  DATA line (SDA)
P1.2  ->  CLOCK line (SCL)

As the hardware implementation of the serial EEPROM module manager circuit, the slave address signals A0 and A1 are connected to a logical 0, while signal A2 is connected to logical 1. Logic state 0 corresponds to low level (≈0 V), logic state 1 corresponds to high level (≈5 V). During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the two pins of EEPROM are set to high.

BACKED SRAM + SERIAL RTC

For software management of serial SRAM+RTC module of IC 9, please refer to specific documentation or to demo programs supplied with the card. The User must realize a serial communication with I2C bus standard protocol, through two I/O microprocessor pins. The only necessary information is the electric connection:

P1.3  <->  DATA line (SDA)
P1.2  ->  CLOCK line (SCL)

As the hardware implementation of the serial EEPROM module manager circuit, the slave address signal A0 is connected to a logical 0. Logic state 0 corresponds to low level (≈0 V), logic state 1 corresponds to high level (≈5 V). During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the two pins of SRAM+RTC are set to high.
DIGITAL INPUTS

The status of the 9 NPN digital inputs is acquired through the driver serializer 74 HCT 166 (inputs IN0÷7) or directly through a microcontroller I/O pin (input AUX), with the following correspondance:

- Line H 74 HCT 166 <- Input IN0
- Line G 74 HCT 166 <- Input IN1
- Line F 74 HCT 166 <- Input IN2
- Line E 74 HCT 166 <- Input IN3
- Line D 74 HCT 166 <- Input IN4
- Line C 74 HCT 166 <- Input IN5
- Line B 74 HCT 166 <- Input IN6
- Line A 74 HCT 166 <- Input IN7
- Pin P3.2 (/INT0) CPU <- Input AUX

When an NPN input is activated (contact closed to the GND opto signal) the corresponding line is at a low logic state (logic 0), vice versa when an input is not activated (contact open) a high logic level is acquired (logic 1).

As previously stated, LEDs LD6÷12 give the User a visual indication of the digital inputs status (LED on=input activated).

For informations about the management of the 74HCT166 serializer (IC1) please refer to the manufacturer documentation. This technical manual reports no software information about this component because its management requires a deep knowledge of it and anyway the user can take advantage of the special high level procedures delivered with the programming package. Control logic allows the software management of this microprocessor through three I/O microprocessor pins, with the following correspondance:

- P3.3 <- DATA signal
- P3.4 -> LOAD signal
- P3.5 -> signal CLK

Logic state 0 corresponds to low level (=0 V), logic state 1 corresponds to high level (=5 V). During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the pins of 74 HCT 166 are set to high.

As an example, here is reported a procedure written for BASCOM 8051 compiler, that returns the status of digital inputs IN0÷7 through 74 HCT 166.

```
' Constants declaration
Dt_166 Alias P3.3 ' 74 HCT 166 DATA signal
Ld_166 Alias P3.4 ' 74 HCT 166 LOAD signal
Clk_166 Alias P3.5 ' 74 HCT 166 CLK signal

' Variables declaration
Dim In0_7 As Byte ' Status of inputs IN0..IN7
Dim I As Byte ' General purpose counter
Dim Flg As Bit ' General purpose flag
```
' Procedures declaration
Declare Sub Init_166
Declare Sub Inp_166

' Procedure: INIT_166
' Initializes 74 HCT 166 management lines
' Input: --
' Output: --

Sub Init_166
  Clk_166 = 1 ' Sets Clock to 1
  Ld_166 = 1 ' Load signal unactivated
  Dt_166 = 1 ' Sets Data signal as input
  Flg = Dt_166
Return

' Procedure: INP_166
' Return the status of the inputs IN0..IN7
' Input: --
' Output: In0_7 = Status of IN0 (bit 0)..IN7 (bit7)

Sub Inp_166
  In0_7 = 0 ' Resets the variable
  Clk_166 = 0 ' Clock starting status
  Ld_166 = 0 ' Must acquire byte in input
  Clk_166 = 1 ' Clock front to acquire data
  Ld_166 = 1
  For I = 1 To 8 ' Loop to acquire 8 bits
    Flg = Dt_166 ' Read one bit
    Clk_166 = 0 ' Sets Clock to 0
    If Flg = 1 Then ' Sets bit In0_7.7 to the status of Flg
      Set In0_7.7
    Else
      Reset In0_7.7
    End If
    If I <> 8 Then
      clr c
      Rotate In0_7 , Right , 1 ' Rights shifts the byte of 1 position
      Clk_166 = 1 ' Clock front for next reading
    End If
  Next I
Return
DIGITAL OUTPUTS

The status of the four digital outputs, realys or NPN transistors, is defined through the management of four microcontroller I/O pins, with the following correspondence:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Output</th>
<th>Correspondence</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1.4</td>
<td>OUT0</td>
<td>relay RL1 or transistor Q1</td>
</tr>
<tr>
<td>P1.5</td>
<td>OUT1</td>
<td>relay RL2 or transistor Q2</td>
</tr>
<tr>
<td>P1.6</td>
<td>OUT2</td>
<td>relay RL3 or transistor Q3</td>
</tr>
<tr>
<td>P1.7</td>
<td>OUT3</td>
<td>relay RL4 or transistor Q4</td>
</tr>
</tbody>
</table>

When the I/O signals are set to logic state low (logic 0), the corresponding output is activated (the transistor is conducting or the relay contact is connected to its common terminal), vice versa when pins are set to logic state high (logic 1), the outputs OUTn are deactivated (the transistor is not conducting or the relay contact is open).

As previously stated, LEDs LD1÷4 give the User a visual indication of the status of the digital outputs (LED on=output activated).

During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the relays or the transistors are deactivated.

JUMPER J1 (RUN/SETUP) - A/D LINE

The status of jumpers J5 and J6 can be acquired by software. In fact when connected it short-circuits the CPU pin P1.1 (AN1) and pin P1.0 (AN0). This feature allows the User to implement two system configurations but only if the above mentioned pins are not used to manage the A/D conversion line, the analog comparator or the I/O TTL available on the GPC® R/T94 board. For example, firmware ALB x94 uses this feature to distinguish between RUN mode and SETUP mode.

Under the point of view of software management, being the procedure to acquire such configuration input quite articulated, in case of need please contact grifo®.

CPU PERIPHERALS

A description of all the microprocessor CPU and registers (Timer Counter, interrupts controller, serial line, I/O ports, etc.) is available in appendix B of this manual. Should such information be insufficient, please refer to the manufacturer documentation.
A/D CONVERTER

For the A/D Converter section, the conversion is performed through the analog comparator and a timer on board of the microcontroller. For the management of their registers, please refer to appendix B of this manual.

About the software management of the conversion, this technical manual reports no software information about this operation because its management requires a deep knowledge, also theoretical, of it and anyway the User can take advantage of the ATMEL application notes, available on Internet or in the Data Book or use the specific procedures provided with high-level languages software packages.

As an example, here is reported a procedure written for **BASCOM 8051** compiler, that returns the digital combination corresponding to the input analog signal itself.

In this example timer 0 is used to generate the time base.

```
' Constants declaration
Tr0 Alias Tcon.4 ' Flag start/stop of TIMER 0
Tf0 Alias Tcon.5 ' Flag Over-flow of TIMER 0
An0 Alias P1.0 ' A/D charge capacity input
An1 Alias P1.1 ' A/D analog input

' Procedures declaration
Declare Sub AD_Conv

' Procedure: AD_CONV
' Returns the conversion acquired by the A/D section in 60 msec max
' Input: --
' Output: TH0, TL0 = Combination acquired

Sub AD_Conv
Tr0 = 0 ' Stops Timer 0
Tf0 = 0 ' Resets Over-flow flag
Th0 = 0 ' Reset Timer 0 registers
Tl0 = 0
An1 = 1 ' Unlocks A/D analog input
An0 = 1 ' Starts to charge the capacitor
Tr0 = 1 ' Starts Timer 0
'Tr0 = 0 ' Assembler routine starts

!Ad_2:
    jb tf0 , ad_3 ; ' Checks conversion end by Over-flow
    jb p3.6 , ad_3 ; ' Checks conversion end by testing if
                   ' bit P3.6 (comparator output) is = 1
Sjmp ad_2
!Ad_3:
'Tr0 = 0 ' Assembler routine ends

Tr0 = 0 ' Stops Timer 0
```
If Tf0 = 1 Then ' If Timer 0 is in Over-flow, sets....
    Th0 = 255 ' ....conversion value to full range
    Ti0 = 255
End If
Tf0 = 0 ' Resets Over-flow flag
An0 = 0 ' Discharges the capacitor for about 6 msec
Waitms 6
End Sub
ALB (ABACO® LINK BUS) SOFTWARE DESCRIPTION

The ALB communication protocol allows to take advantage of all the on board resources by means of a set of commands; all the characters received on the serial communication line are interpreted and executed, then an eventual answer can be transmitted to the master control unit. The firmware features also a Setup mode, which allows the user to configure every section of the device.

WORKING MODE SELECTION

The firmware of GPC® R/T94 board can manage two different working modes, these are Setup mode or Run mode (ALB). The selection of which mode to employ happens during the Power-ON phase, by testing the status of jumpers J5 and J6:

SETUP mode: J5 and J6 in position 2-3 and 4-5
Pin 3-3 of J5 and J6 CONNECTED

RUN mode: J5 and J6 in any position
Pin 3-3 of J5 and J6 NOT CONNECTED

SETUP MODE

In SETUP mode it is possible to set the initialization parameters, that is the baud rate, the communication mode and the device name. These settings will be stored in the EEPROM, and will make the working configuration in RUN mode. To correctly set the initialization parameters please follow the instructions below:

1) Turn off GPC® R/T 94.

2) Be sure that no signal is connected to inputs and outputs.

3) Connect jumpers J5 and J6 to select SETUP mode as described in the previous paragraph.

4) Supply the board.

5) If SETUP mode has been recognized then the output OUT0 has been activated (LD1 is ON). At this point the User can set baud rate and communication mode configuring IN0÷IN7 inputs as described in the following tables.

NOTE
The desired setting must be kept up to the following point.
ON and OFF indicate respectively an input connected and not connected to GND opto signal. To confirm and store the selected parameters, the User must close the AUX input to GND opto signal and reopen it.

6) If the previous operation didn't succeed then also output OUT3 is activated (LD4 ON) and the User must repeat the operation at point 5. If, otherwise, the operation succeeded, the firmware activates output OUT1 (LD2 ON), to tell that it is possible to set the NAME used by the board for Master Slave serial communication. To set the NAME (permitted values are in the range 128 ÷ 255) the same technique described at point 2 should be used, that is, to set an opportune input configuration on signals IN0÷IN7 then to send an impulse to input AUX to confirm the data. Please consider that the NAME is input following a binary codification (input INx in position ON means the corresponding bit set to 0), as reported in the following table, and that the NAME is not significant if the point to point communication mode has been selected.
7) If the previous operation didn't succeed then also output OUT3 is activated (LD4 ON) and the User must repeat the operation at point 6. If, otherwise, the operation succeeded, the firmware stores all the input parameters into the EEPROM and starts an infinite loop. If this last operation didn't succeed all the OUT0÷OUT3 outputs are activated (LD1÷LD4 ON), otherwise the firmware states the end of the SETUP procedure setting the outputs in the following configuration:

<table>
<thead>
<tr>
<th>OUT0</th>
<th>Activated (LD1 ON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT1</td>
<td>Deactivated (LD2 OFF)</td>
</tr>
<tr>
<td>OUT2</td>
<td>Activated (LD3 ON)</td>
</tr>
<tr>
<td>OUT3</td>
<td>Deactivated (LD4 OFF)</td>
</tr>
</tbody>
</table>

8) Turn off power supply

9) Connect jumpers J5 and J6 to select RUN mode.
RUN MODE

When entering in RUN mode the baud rate and communication protocol parameters stored in EEPROM are verified. If they are not valid (e.g. EEPROM not initialized) the board starts a standby loop; at this point the User may only turn off the device.

NOTE

EEPROM is NOT initialized by default, the the user must initialize it (SETUP mode) before attempting to use the board.

Baud rate and communication protocol for the RUN mode are settable (in SETUP mode), while data format is function of the selected communication mode, as follows:

Point to Point Communication: 8 bit, 1 Stop, NO Parity
Master Slave: 9 bit, 1 Stop, NO Parity

In RUN mode the board's peripherals are managed in the following modalities.

Serial line:
Used to receive commands and send eventual responses; it is possible to communicate in point to point or master slave mode. This last mode allows to make a remote control network where the single modules are driven by an unique master unit (PC, PLC, GPC® card, etc.) also from great distance.

Digital outputs OUT0÷OUT3:
They all can be set at the same time by an unique command, or they can be managed bit level. In this case there are also commands to make timed activations, square waves, etc.

Digital inputs IN0÷IN7:
They are acquired at the same time by an unique command, or they can be managed bit level.

Digital input AUX:
It can be used as digital input line, or as a clock signal for a 16 bits counter, managed through specific commands.

Serial EEPROM:
A set of commands that allow to manage the content of the serial EEPROM as 10 characters long messages is available. So becomes possible to store in non volatile memory informations like text strings, settings, parameters, etc.

SRAM+RTC:
A set of commands that allow to set hours, minutes, seconds, day, month, year and day of week of the Real Time Clock is available. It is also possible to read and write bytes of the SRAM that this device provides.

TTL I/O signals:
They implement two different 1-Wire® BUS protocol lines. Also, a set of commands to manage the devices connectable to such BUS (temperature sensor, memories, Dallas iButton™, etc.) is available.
GENERAL PURPOSE COMMANDS

Here follows the general purpose commands list.

MASTER RESET

Input Sequence:
- Dec Code: 65 97
- Hex Code: 41 61
- Mnemonic: Aa

Upon the reception of this command the firmware restores the initial condition that happens to be at the Power-ON; in detail:

- OUT0÷OUT3: They are reset and put into initial condition, then they are set to the logic state 0. Eventual timings are interrupted.
- 16 bits Counter: It is initialized to 0.

READ FIRMWARE VERSION

Input Sequence:
- Dec Code: 121
- Hex Code: 79
- Mnemonic: y

Answer Codes:
The two nibbles of the firmware version are returned:

\[ \text{Version } X.Y = Y \rightarrow \text{<NibL VAL>} \text{ <NibH VAL>} \]

CHECK EEPROM READY FOR WRITING

Input Sequence:
- Dec Code: 66
- Hex Code: 42
- Mnemonic: B

By this command the user may ask the firmware if it is ready to manage a new EEPROM message, this command must be sent whenever the User needs to send one of the following messages management commands.

Answer Codes:
The firmware returns the following codes:

\[ 0 \quad \text{EEPROM not ready to manage a new message} \]
\[ 1 \quad \text{EEPROM ready to manage a new message} \]
PRESENCE BYTE OUTPUT

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code:</th>
<th>dat.L</th>
<th>dat.H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex Code:</td>
<td>dat.L</td>
<td>dat.H</td>
</tr>
<tr>
<td>Mnemonic:</td>
<td>Y</td>
<td>ASCII(dat.L)</td>
</tr>
</tbody>
</table>

The presence byte of the board is set to the value of the "dat" parameter, which can range in the values 0÷255.

This byte is allocated in a reserved EEPROM location and, once it has been set, allows to, for example, verify the correct working of GPC® R/T 94 or check for conflicts on the serial communication line.

Please remark that "dat" parameter must be sent as two nibbles, in detail must be sent the low nibble first (dat.L=bits 0÷3), then the high nibble (dat.H=bits 4÷7).

NOTE
This command requires to write into the on board serial EEPROM, so before attempting to execute it the User shold be sure that such device is ready to be written in, if this condition is not satisfied the command is ignored.

Example:
If the User wishes to store the presence byte "65" then he/she will need to send the sequence: 89 1 4.

PRESENCE BYTE INPUT

Input Sequence:

| Dec Code: | 121 |
| Hex Code: | 79  |
| Mnemonic: | y   |

GPC® R/T 94 sends to the serial port its presence byte.
This command is useful to verify the correct working of GPC® R/T 94 or check for conflicts on the serial communication line.

Answer Codes:
A value ranging from 0 to 255 is returned by sending two nibbles in the format seen for the previous command: low nibble first (dat.L=bits 0÷3), then the high nibble (dat.H=bits 4÷7).
AUX SIGNAL CONFIGURATION SETTING

Input Sequence:

- **Dec Code:** 85  
- **Hex Code:** 55  
- **Mnemonic:** U ASCII(cfg)

The AUX signal may be configured as a general purpose INPUT or as a trigger input for the 16 bits internal counter according to the value of \( \text{cfg} \):

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>AUX signal is set as an INPUT</td>
</tr>
<tr>
<td>2</td>
<td>AUX signal is used to Trigger the 16 bit Counter</td>
</tr>
</tbody>
</table>

If the sequence contains invalid data the command is ignored.

**NOTE**
This command requires to write into the on board EEPROM, so before attempting to execute it the User shold be sure that such device is ready to be written in, if this condition is not satisfied the command is ignored.

**Example:**
If the User wants to set AUX signal as COUNTER then he/she will need to send the following sequence: 85 2.

AUX SIGNAL CONFIGURATION ACQUISITION

Input Sequence:

- **Dec Code:** 102  
- **Hex Code:** 66  
- **Mnemonic:** f

Answer Codes:
The AUX signal configuration byte returned by the firmware may assume one of the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>AUX signal is set as an INPUT</td>
</tr>
<tr>
<td>2</td>
<td>AUX signal is used to Trigger the 16 bit Counter</td>
</tr>
</tbody>
</table>

READ BOARD CODE

Input Sequence:

- **Dec Code:** 75  
- **Hex Code:** 4B  
- **Mnemonic:** K

Answer Codes:
The value 1, which is the GPC® R/T 94 identificative code, is returned.
DIGITAL I/O PORT BYTE MANAGEMENT COMMANDS

Here follow the commands to manage byte-level the digital I/O ports of GPC® R/T 94.

OUTPUT PORT SET

Input Sequence:

Dec Code: 87 1 dat 0  
Hex Code: 57 1 dat 0  
Mnemonic: W SOH ASCII(dat) NUL

The <Data> byte must be sent according to the following format:

(MSB) 0 0 0 0 OUT3 OUT2 OUT1 OUT0 (LSB)

Where OUTn stands for the logic state, 0 (output disabled) or 1 (output activated), that the respective output must get.
If the sequence contains invalid data the command is ignored.

Example:
If the User wants to activate the OUT0 and OUT3 outputs then he/she will need to send the following sequence:
87 1 9 0.

INPUT PORT ACQUISITION

Input Sequence:

Dec Code: 82 0  
Hex Code: 52 0  
Mnemonic: R NUL

The serial data read by the input port is returned.

Answer Codes:
The data acquired by the port is returned as two nibbles in the following format:

Nibble L: (MSB) 0 0 0 0 IN3 IN2 IN1 IN0 (LSB)  
Nibble H: 0 0 0 0 IN7 IN6 IN5 IN4

Where INn stands for the logic state, 0 (ON = input connected to GND opto) or 1 (OFF = input open), that the respective NPN lines have.
If the sequence contains invalid data the command is ignored.

Example:
If the User wants to read the input port, where the 90 (5A Hex) data is present then he/she will need to send the following sequence: 82 0, obtaining the following answer: 10 5.
DIGITAL I/O PORT BIT MANAGEMENT COMMANDS

Here follow the commands to manage bit-level the digital I/O ports of GPC® R/T 94.

OUTPUT BIT SET

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>83</td>
<td>53</td>
<td>S SOH ASCII(bit)</td>
</tr>
</tbody>
</table>

The OUTn output indicated by bit gets the logic state 1 (output activated); bit can be 0, 1, 2 or 3. Eventual timings occurring on the output line are interrupted. If the sequence contains invalid data the command is ignored.

Example:
If you want to activate the OUT2 output you will need to send the following sequence:

83 1 2.

TIMED OUTPUT BIT SET

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>115</td>
<td>73</td>
<td>s SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)</td>
</tr>
</tbody>
</table>

The OUTn output indicated by bit gets the logic state 1 (line activated); bit can be 0, 1, 2 or 3. The selected output holds the logic state 1 for a number of cycles (1 cycle = 10 msec) determined by the tmp byte, then it returns to the logic state 0 (line disabled). The timing value must range 1÷255 and must be sent as nibbles in detail must be sent the low nibble first (tmp.L=bits 0÷3), then the high nibble (tmp.H=bits 4÷7).

If the sequence contains invalid data the command is ignored.

Example:
If the User want to activate the OUT2 output for 500 msec, corresponding to 50 cycles, then he/she will need to send the following sequence:

115 1 2 2 3.
OUTPUT BIT CLEAR

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>67</td>
<td>43</td>
<td>C SOH ASCII(bit)</td>
</tr>
</tbody>
</table>

The OUTn output indicated by bit gets the logic state 0 (output disabled); bit can be 0, 1, 2 or 3. Eventual timings occurring on the output line are interrupted. If the sequence contains invalid data the command is ignored.

Example:
If you want to deactivate the OUT2 output, you will need to send the following sequence:
67 1 2.

TIMED OUTPUT BIT CLEAR

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>63</td>
<td>c SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)</td>
</tr>
</tbody>
</table>

The OUTn output indicated by bit gets the logic state 0 (line disabled); bit can be 0, 1, 2 or 3. The selected output holds the logic state 0 for a number of cycles (1 cycle = 10 msec) determined by the tmp byte, then it returns to the logic state 1 (line activated). The timing value must range 1÷255 and must be sent as nibbles in detail must be sent the low nibble first (tmp.L=bits 0÷3), then the high nibble (tmp.H=bits 4÷7). If the sequence contains invalid data the command is ignored.

Example:
If the User wants to deactivate the OUT2 output for 500 msec, that is 50 cycles, then he/she will need to send the following sequence:
99 1 2 2 3.
INPUT BIT OR AUX INPUT ACQUISITION

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
<th>port</th>
<th>bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>114</td>
<td>72</td>
<td>r ASCII(port)</td>
<td>ASCII(bit)</td>
<td></td>
</tr>
</tbody>
</table>

The logic state of the digital NPN input indicated by the parameters port and bit is acquired and returned as follows:

- Input IN0 ÷ IN7 -> port = 0 bit = 0 ÷ 7
- Input AUX -> port = 3 bit = 0

If the sequence contains invalid data the command is ignored.

Answer codes:
A byte reporting logic state of the digital NPN input indicated is returned; in particular the value 0 (input connected to GND opto) or 1 (input open) can be returned.

Example:
If the User wants to read the AUX input, then he/she will need to send: 114 3 0

DEBOUNCED READ OF AN INPUT PORT BIT

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
<th>port</th>
<th>bit</th>
<th>tmp.L</th>
<th>tmp.H</th>
</tr>
</thead>
<tbody>
<tr>
<td>68</td>
<td>44</td>
<td>D ASCII(port)</td>
<td>ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The logic state of the digital NPN input indicated by the parameters port and bit is acquired and returned as follows:

- Input IN0 ÷ IN7 -> port = 0 bit = 0 ÷ 7
- Input AUX -> port = 3 bit = 0

Differently from the previous command, the acquisition here is performed through a debouncing process whose duration is specified by parameter tmp and expressed in cycles (1 cycle = 10 msec). The timing value must range 1 ÷ 255 and must be sent as nibbles, in detail must be sent the low nibble first (tmp.L=bits 0 ÷ 3), then the high nibble (tmp.H=bits 4 ÷ 7).

If the sequence contains invalid data the command is ignored.

Answer codes:
The data returned may assume one of the following values:

- 0: Input closed to GND opto during the whole debouncing
- 1: Input open during the whole debouncing
- 7: Input changed its status during debouncing

Example:
If the User wants to read the IN2 input with a debouncing time of 50 msec, corresponding to 5 cycles, then he/she will need to send the following sequence: 68 0 2 5 0
SQUARE WAVE STARTING WITH "1" OUTPUT BIT

Input Sequence:

Dec Code: 112 1 bit tmp.L tmp.H
sta.L sta.H

Hex Code: 70 1 bit tmp.L tmp.H
sta.L sta.H

Mnemonic: p SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)
ASCII(sta.L) ASCII(sta.H)

The OUTn output indicated by bit outputs a square wave with 50% of Duty Cycle starting by a logical state "1"; bit can be 0, 1, 2 or 3. The the half-period of the signal contains a number of cycles (1 cycle = 10 msec) determined by the tmp byte. The timing value must range 1÷255 and must be sent as nibbles: in detail must be sent the low nibble first (tmp.L=bits 0÷3), then the high nibble (tmp.H=bits 4÷7).

The number of commutations that the signal must perform is indicated by the parameter sta; as shown by the picture sta+1 commutations are performed. Also this value must range 1÷255 and must be sent as nibbles: in detail must be sent the low nibble first (sta.L=bits 0÷3), then the high nibble (sta.H=bits 4÷7).

If the sequence contains invalid data the command is ignored.

Example:
If the User want to activate the OUT2 output for 200 msec, corresponding to 20 cycles and making it commutate 10 times, then he/she will need to send the following sequence:

112 1 2 4 1 9 0.

NOTE
As can be seen by the picture, the final status of the input will depend on the number of commutations performed; in particular an even number of commutations will leave the output in status 0 (output disabled) and viceversa.
SQUARE WAVE STARTING WITH "0" OUTPUT BIT

Input Sequence:

- **Dec Code:** 119 1
  - bit
  - tmp.L
  - tmp.H
  - sta.L
  - sta.H
- **Hex Code:** 77 1
  - bit
  - tmp.L
  - tmp.H
  - sta.L
  - sta.H
- **Mnemonic:** w SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H) ASCII(sta.L) ASCII(sta.H)

**Figure 44: Timed Square Wave Command**

The OUTn output indicated by bit outputs a square wave with 50% of Duty Cycle starting by a logical state "0"; bit can be 0, 1, 2 or 3. The half-period of the signal contains a number of cycles (1 cycle = 10 msec) determined by the tmp byte. The timing value must range 1\(\div\)255 and must be sent as nibbles: in detail must be sent the low nibble first (tmp.L=bits 0\(\div\)3), then the high nibble (tmp.H=bits 4\(\div\)7).

The number of commutations that the signal must perform is indicated by the parameter sta; as shown by the picture sta+1 commutations are performed. Also this value must range 1\(\div\)255 and must be sent as nibbles: in detail must be sent the low nibble first (sta.L=bits 0\(\div\)3), then the high nibble (sta.H=bits 4\(\div\)7).

If the sequence contains invalid data the command is ignored.

**NOTE**

As can be seen by the picture, the final status of the input will depend on the number of commutations performed; in particular an even number of commutations will leave the output in status 1 (output disabled) and viceversa.

**Example:**
If the User wants to deactivate the OUT2 output for 200 msec, corresponding to 20 cycles and making it commutate 10 times, then he/she will need to send the following sequence:

119 1 2 4 1 9 0.
SQUARE WAVE OUTPUT BIT

Input Sequence:

- **Dec Code:** 80 1 bit tmp.L tmp.H
- **Hex Code:** 50 1 bit tmp.L tmp.H
- **Mnemonic:** P SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)

![Figure 45: SQUARE WAVE COMMAND](image)

The OUTn output indicated by bit outputs a square wave with 50% of Duty Cycle; bit can be 0, 1, 2 or 3. The half-period of the signal contains a number of cycles (1 cycle = 10 msec) determined by the tmp byte. The timing value must range 1÷255 and must be sent as nibbles: in detail must be sent the low nibble first (tmp.L=bits 0÷3), then the high nibble (tmp.H=bits 4÷7).

If the sequence contains invalid data the command is ignored.

Example:
If the User wants to deactivate the OUT2 output for 200 msec, that is 20 cycles, then he/she will need to send the following sequence: 99 1 2 4 1.
16 BIT COUNTER MANAGEMENT COMMANDS

Here follow the commands to manage the 16 bit counter. Its value is incremented by the commutations of the AUX signal when this is configured to trigger the counter.

16 BIT COUNTER READ

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>73</td>
<td>49</td>
<td>I</td>
</tr>
</tbody>
</table>

This command allows to acquire the current value of the 16 bit counter.

Answer Codes:
The sequence returned by the command is made of four bytes showing the 16 bit value currently stored in the counter register; this is sent as nibbles according to the following format:

\[
\begin{array}{cccccccc}
\text{Counter} & (\text{bit 0÷3}) & : & (\text{MSB}) & 0 & 0 & 0 & 0 \\
& (\text{bit 4÷7}) & : & 0 & 0 & 0 & 0 & \text{Bit3} \\
& (\text{bit 8÷11}) & : & 0 & 0 & 0 & 0 & \text{Bit7} \\
& (\text{bit 12÷15}) & : & 0 & 0 & 0 & 0 & \text{Bit15}
\end{array}
\]

When the counter reaches the maximum value, which equals to 65535 (FFFF Hex), the next trigger impulse will set the counter to 0. If the AUX signal is configured as INPUT this command will always return 0.

Example:
If the counter register contains the value 23055 (5A0F Hex), sending the command 73 will return the following values: 15 0 10 5.

16 BIT COUNTER RESET

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Hex Code</th>
<th>Mnemonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>88 120</td>
<td>58 78</td>
<td>X x</td>
</tr>
</tbody>
</table>

Upon the reception of this command the firmware resets the 16 bit counter, which will carry the new value 0.
MESSAGES MANAGEMENT COMMANDS

LAST MEMORIZABLE MESSAGE ACQUISITION

Input Sequence:

Dec Code: 77
Hex Code: 4D
Mnemonic: M

This command allows the User to know the maximum number of messages that the board can store. This number depends on the memory device installed according to the following table:

<table>
<thead>
<tr>
<th>EEPROM</th>
<th>N.MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>24c02 (256 Bytes)</td>
<td>23</td>
</tr>
<tr>
<td>24c04 (512 Bytes)</td>
<td>48</td>
</tr>
<tr>
<td>24c08 (1024 Bytes)</td>
<td>99</td>
</tr>
</tbody>
</table>

**Figure 46: Maximum number of messages memorizable in EEPROM**

Answer Codes:
The number is returned in two nibbles: in particular first the low nibble will be sent \( \text{carn.L} = \text{bit } 0 \div 3 \), then the high nibble \( \text{carn.H} = \text{bit } 4 \div 7 \).

STORING A MESSAGE

Input Sequence:

Mnemonic: E ASCII(msg.L) ASCII(msg.H) ASCII(car0.L) ASCII(car0.H)...ASCII(car9.L) ASCII(car9.H)

The ten characters long message, whose code is indicated by \( \text{msg} \), is stored in EEPROM. The message number must range \( 0 \div \text{N.MAX} \) and must be sent in two nibbles: in detail must be sent the low nibble first \( \text{msg.L} = \text{bits } 0 \div 3 \), then the high nibble \( \text{msg.H} = \text{bits } 4 \div 7 \).

The value of N.MAX may be acquired by the apposite command.
The characters codes must be in the range $0 \div 255$ ($0 \div FF$ Hex) and must be sent in nibbles: in detail must be sent the low nibble first ($\text{carn.L}$=bits 0÷3), then the high nibble ($\text{carn.H}$=bits 4÷7).

If the sequence contains invalid data the command is ignored.

**NOTE**
This command requires to write into the on board serial EEPROM, so before attempting to execute it the User should be sure that such device is ready to be written in, if this condition is not satisfied the command is ignored.

**Example:**
If the User wants to store the message "ABCDEFGHIJ" (corresponding to the codes: 65, 66, 67, 68, 69, 70, 71, 72, 73, 74) with number 16, then he/she will need to send the following sequence:

$$
69 \ 0 \ 1 \ 1 \ 4 \ 2 \ 4 \ 3 \ 4 \ 4 \ 4 \ 5 \ 4 \ 6 \ 4 \ 7 \ 4 \ 8 \ 4 \ 9 \ 4 \ 10 \ 4.
$$

**READING A MESSAGE**

**Input Sequence:**

- *Dec Code:* 76  msg.L  msg.H
- *Hex Code:* 4C  msg.L  msg.H
- *Mnemonic:* L  ASCII(msg.L)  ASCII(msg.H)

The ten characters long message, whose code is indicated by $\text{msg}$, is read from the EEPROM and sent on the serial connection. The message number must range $0 \div \text{N.MAX}$ and must be sent in two nibbles: in detail must be sent the low nibble first ($\text{msg.L}$=bits 0÷3), then the high nibble ($\text{msg.H}$=bits 4÷7).

The value of N.MAX may be acquired by the apposite command.

If the sequence contains invalid data the command is ignored.

**Answer Codes:**
The ten characters are returned in nibbles: in detail must be sent the low nibble first ($\text{carn.L}$=bits 0÷3), then the high nibble ($\text{carn.H}$=bits 4÷7).

If the User wants to read the message with number 16 stored in the previous example, then he/she needs to send the following sequence: 7601. The answer will be the sequence:

$$
1 \ 4 \ 2 \ 4 \ 3 \ 4 \ 4 \ 4 \ 5 \ 4 \ 6 \ 4 \ 7 \ 4 \ 8 \ 4 \ 9 \ 4 \ 10 \ 4.
$$
SRAM+RTC MANAGEMENT COMMANDS

CLOCK SETTING

Input Sequence:

           gse.L gse.H

           gse.L gse.H

Mnemonic: O ASCII(ore.L) ASCII(ore.H) ASCII(min.L) ASCII(min.H)
           ASCII(sec.L) ASCII(sec.H) ASCII(gio.L) ASCII(gio.H)
           ASCII(mes.L) ASCII(mes.H) ASCII(ann.L) ASCII(ann.H)
           ASCII(gse.L) ASCII(gse.H)

The Real Time Clock registers are initialized as follows: hours with ore, minutes with min, seconds with sec, day with gio, month with mes, year with ann and day of week with gse.
Each of these parameters must be sent in two nibbles: in detail must be sent the low nibble first (xxx.L=bits 0÷3), then the high nibble (xxx.H=bits 4÷7). The value must be in the respective range as shown in the following table:

<table>
<thead>
<tr>
<th>BYTE</th>
<th>RANGE</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORE</td>
<td>0 ... 23</td>
<td>HOURS</td>
</tr>
<tr>
<td>MIN</td>
<td>0 ... 59</td>
<td>MINUTES</td>
</tr>
<tr>
<td>SEC</td>
<td>0 ... 59</td>
<td>SECONDS</td>
</tr>
<tr>
<td>GIO</td>
<td>1 ... 31</td>
<td>DAY</td>
</tr>
<tr>
<td>MES</td>
<td>1 ... 12</td>
<td>MONTH</td>
</tr>
<tr>
<td>ANN</td>
<td>0 ... 99</td>
<td>YEAR</td>
</tr>
</tbody>
</table>
| GSE  | 0 ... 6    | Day of week: 0 -> SUNDAY
                   6 -> SATURDAY

Figure 47: RTC initialization bytes validity range

If the sequence contains invalid data the command is ignored.

Example:
If the User wishes to set the RTC as: Monday May 11th 1998 12:30:40 the nhe/she will need to send:
79 12 0 14 1 8 2 11 0 5 0 2 6 1 0.
CLOCK READ

**Input Sequence:**

- **Dec Code:** 111
- **Hex Code:** 6F
- **Mnemonic:** o

The Real Time Clock registers are read and their content is sent to the serial port.

**Answer Codes:**

Seven bytes representing hours, minutes, seconds, day, month, year and day of week are returned, following the format shown in the previous table. Each of these bytes is sent in two nibbles: in detail the low nibble is sent first (xxx.L=bits 0÷3), then the high nibble is sent (xxx.H=bits 4÷7).

RTC SRAM WRITE

**Input Sequence:**

- **Dec Code:** 71 ind.L ind.H dat.L dat.H
- **Hex Code:** 47 ind.L ind.H dat.L dat.H
- **Mnemonic:** G ASCII(ind.L) ASCII(ind.H) ASCII(dat.L) ASCII(dat.H)

The data **dat** (ranging in 0÷255) is stored in the RTC serial SRAM at the address **ind** (ranging in 32÷255). Each of these parameters must be sent in two nibbles: in detail must be sent the low nibble first (xxx.L=bits 0÷3), then the high nibble (xxx.H=bits 4÷7).

If the sequence contains invalid data the command is ignored.

**Example:**

If the User wishes to store data "65" at the address "100" then he/she will need to send the sequence: 71 4 6 1 4.

RTC SRAM READ

**Input Sequence:**

- **Codice Dec:** 103 ind.L ind.H
- **Codice Hex:** 67 ind.L ind.H
- **Mnemonic:** g ASCII(ind.L) ASCII(ind.H)

The data stored in the RTC serial SRAM at the address **ind** (ranging 32÷255) is read and sent to the serial port. The parameter **ind** is the address to read from. It must be sent in two nibbles: in detail must be sent the low nibble first (ind.L=bits 0÷3), then the high nibble (ind.H=bits 4÷7).

If the sequence contains invalid data the command is ignored.

**Codici di Risposta:**

The data byte, ranging in 0÷255, is returned. It is sent in two nibbles: in detail the low nibble is sent first (dat.L=bits 0÷3), then the high nibble is sent (dat.H=bits 4÷7).
1-WIRE® COMMUNICATION BUS LINES

The TTL I/O lines available on GPC® R/T 94 are used to implement a 1-Wire® BUS communication protocol. By means of the commands explained in the following paragraph it is possible to manage the several devices developed to work with this standard (temperature sensors, memories, Dallas iButton™, etc.).

The TTL I/O signals available on connector CN6 are used as follows:

- **Pin 1 CN6** -> Line BUS 1-Wire® n. 0
- **Pin 2 CN6** -> Line BUS 1-Wire® n. 1

As can be seen in the following paragraph, the high level commands available do not support the presence of more than one device on each BUS, in fact, for example, there is not the command “Search ROM”, whose task is to search for ROM codes on the line.

It is possible, anyway, to manage a network connection of many 1-Wire® units taking advantage of the low level commands (BUS reset, get and set a bit status, read and write bytes); in this case the implementation happens to be very articialte and requires also a heavy communication between the master unit and GPC® R/T 94.

It is suggestable to connect the board to at most two 1-Wire® devices; the following figure reports a connection example with two temperature sensors Dallas DS18s20.

![Connection Diagram](image)

**FIGURE 48: CONNECTION EXAMPLE WITH TWO 1-WIRE® DEVICES**

**NOTE**

For a correct use of the two 1-Wire® lines, jumpers J5 and J6 of GPC® R/T 94 must be in position 1-2 and 3-4, to configure the pins of CN6 as generic I/O lines.
COMMANDS TO MANAGE THE 1-WIRE® BUS LINES

Here follow the commands used to manage the two 1-Wire® BUS lines available on GPC® R/T 94.

RESET 1-WIRE® BUS AND GET PRESENCE PULSE

**Input Sequence:**

<table>
<thead>
<tr>
<th>Dec Code:</th>
<th>Hex Code:</th>
<th>Mnemonic:</th>
</tr>
</thead>
<tbody>
<tr>
<td>33 85</td>
<td>21 55</td>
<td>! U ASCII(wire)</td>
</tr>
</tbody>
</table>

The reset sequence is sent to the 1-Wire® BUS line indicated by parameter `wire`. The value of `wire` must be 0 or 1, otherwise the command is ignored.

After the reset sequence has finished, the presence impulse of an eventual other device on the line is acquired.

**Answer Codes:**
The presence impulse logic status is returned, in detail:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1-Wire® device present and ready to receive commands</td>
</tr>
<tr>
<td>1</td>
<td>1-Wire® device not present</td>
</tr>
</tbody>
</table>

**Example:**

If the User wants to send the reset sequence and acquire the presence pulse on 1-Wire® BUS n. 1, then he/she will need to send the following sequence: 33 85 1

WRITE A BIT TO 1-WIRE® BUS

**Input Sequence:**

<table>
<thead>
<tr>
<th>Dec Code:</th>
<th>Hex Code:</th>
<th>Mnemonic:</th>
</tr>
</thead>
<tbody>
<tr>
<td>33 119</td>
<td>21 77</td>
<td>! w ASCII(wire) ASCII(bit)</td>
</tr>
</tbody>
</table>

The bit value specified by parameter `bit` is sent to the 1-Wire® BUS line indicated by parameter `wire`. The values of `wire` and `bit` must be 0 or 1, otherwise the command is ignored.

**Example:**

If the User wants to send the bit value 1 on 1-Wire® BUS n. 0, then he/she will need to send the following sequence: 33 119 0 1
READ A BIT FROM 1-WIRE® BUS

Input Sequence:
- **Dec Code:** 33 114 wire
- **Hex Code:** 21 72 wire
- **Mnemonic:** ! r ASCII(wire)

The bit value of the 1-Wire® BUS line indicated by parameter wire is acquired. The value of wire must be 0 or 1, otherwise the command is ignored.

Answer Codes:
The logic status (0 or 1) of the bit acquired from the specified 1-Wire® BUS line is returned.

Example:
If the User wants to read the status of 1-Wire® BUS n. 0, then he/she will need to send the following sequence: 33 114 0

WRITE N BYTES TO 1-WIRE® BUS

Input Sequence:
- **Dec Code:** 33 87 wire n
dat0.L dat0.H ... datn.L datn.H
- **Hex Code:** 21 57 wire n
dat0.L dat0.H ... datn.L datn.H
- **Mnemonic:** ! W ASCII(wire) ASCII(n)
  ASCII(dat0.L) ASCII(dat0.H)...ASCII(datn.L) ASCII(datn.H)

The “n” bytes specified by parameter n are sent to the 1-Wire® BUS line indicated by parameter wire. The value of wire must be 0 or 1; the maximum number of bytes to send after parameter n is 10, so its value must be in the range 1÷10. The “n” bytes value must range 0÷255 and must be sent as nibbles, in detail must be sent the low nibble first (datn.L=bits 0÷3), then the high nibble (datn.H=bits 4÷7). Command is ignored if the sequence contains invalid data.

Example:
If the User wants to send the bytes 23, 118, 80 and 12 to 1-Wire® BUS n. 1, then he/she will need to send the following sequence: 33 87 1 4 7 1 6 7 0 5 12 0

WRITE N BYTES TO 1-WIRE® BUS

Input Sequence:
- **Dec Code:** 33 87 wire n
- **Hex Code:** 21 57 wire n
- **Mnemonic:** ! R ASCII(wire) ASCII(n)

The “n” bytes specified by n are read from the 1-Wire® BUS line indicated by parameter wire.
The value of wire must be 0 or 1; the maximum number of bytes readable with one command is 10, so value of parameter n must be in the range 1±10. Command is ignored if the sequence contains invalid data.

Answer Codes:
The n bytes read from the 1-Wire® BUS line indicated are returned as nibbles, in detail will be received the low nibble first (datx.L=bits 0÷3), then the high nibble (datx.H=bits 4÷7).

Example:
If the User wants to read 4 bytes (that will be 23, 118, 80 and 12) from 1-Wire® BUS number 0, then he/she will need to send the following sequence: 33 8 0 4.
The response will be: 7 1 6 7 0 5 12 0

READ ROM CODE FROM 1-WIRE® BUS

Input Sequence:

<table>
<thead>
<tr>
<th>Dec Code:</th>
<th>Hex Code:</th>
<th>Mnemonic:</th>
</tr>
</thead>
<tbody>
<tr>
<td>33 76 wire</td>
<td>21 4C wire</td>
<td>! L ASCII(wire)</td>
</tr>
</tbody>
</table>

This command will perform the following operations on the 1-Wire® BUS line specified by parameter wire.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receive next command.

b) If the device is present, the Read ROM command (code 33 Hex of 1-Wire® BUS protocol) is sent, then the device ROM code is acquired.

The value of wire must be 0 or 1, otherwise the command is ignored.

Answer Codes:
The response to this command is made of 8 bytes (rom0÷rom7) sent as nibbles, in detail will be received the low nibble first (romx.L=bits 0÷3), then the high nibble (romx.H=bits 4÷7).

Here is the meaning of the bytes:

Device 1-Wire® present and command executed successfully:
The 8 bytes are the device ROM code: rom0=Family code, rom1÷rom6=Serial number and rom7=CRC.

Device 1-Wire® not present and command not sent:
The 8 bytes are all 0's.

Example:
If the User wants to acquire the ROM code of the device connected to 1-Wire® BUS number 1, then he/she will need to send the following sequence: 33 76 0. If the ROM code is: 16 (family code), 56, 198, 13, 0, 8, 0, 226 (CRC), the response to the command will be:
0 1 8 3 6 12 13 0 0 0 8 0 0 0 2 14
MATCH ROM COMMAND ON 1-WIRE® BUS

Input Sequence:

Dec Code: 33 77 wire
rom0.L  rom0.H ... rom7.L  rom7.H

Codice Hex: 21 4D wire
rom0.L  rom0.H ... rom7.L  rom7.H

Mnemonico: ! M ASCII(wire)
ASCII(rom0.L) ASCII(rom0.H)...ASCII(rom7.L) ASCII(rom7.H)

This command will perform the following operations on the 1-Wire® BUS line specified by parameter wire.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receive next command.

b) If the device is present, the Match ROM command (code 55 Hex of 1-Wire® BUS protocol) is sent, then the device ROM code follows: rom0=Family code, rom1÷rom6=Serial number, rom7=CRC.

The value of wire must be 0 or 1, while the 8 bytes “rom0”÷“rom7” may vary in the range 0÷255 and must be sent as nibbles, in detail must be sent the low nibble first (romx.L=bits 0÷3), then the high nibble (romx.H=bits 4÷7).

The command is ignored if the sequence contains invalid data.

Answer Codes:
The response to this command can be:

0 Device 1-Wire® present and command sent
1 Device 1-Wire® not present and command not sent

Example:
If the User wants to send the Match ROM command to 1-Wire® BUS number 1 to match a ROM code: 16 (family code), 56, 198, 13, 0, 8, 0, 226 (CRC), then he/she will have to send the following sequence: 33 77 1 0 1 8 3 6 12 13 0 0 0 8 0 0 0 2 14

SKIP ROM COMMAND ON 1-WIRE® BUS

Input Sequence:

Dec Code: 33 83 wire

Codice Hex: 21 53 wire

Mnemonico: ! S ASCII(wire)

This command will perform the following operations on the 1-Wire® BUS line specified by parameter wire.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receive next command.

b) If the device is present, the Skip ROM command (code CC Hex of 1-Wire® BUS protocol) is sent.
The value of `wire` must be 0 or 1, or the command is ignored.

**Answer Codes:**
The response to this command can be:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Device 1-Wire® present and command sent</td>
</tr>
<tr>
<td>1</td>
<td>Device 1-Wire® not present and command not sent</td>
</tr>
</tbody>
</table>

**Example:**
If the User wants to send the Skip ROM command to 1-Wire® BUS number 1, then he/she will have to send the following sequence: 33 83 1

### ALARM SERCH COMMAND ON 1-WIRE® BUS

**Input Sequence:**

<table>
<thead>
<tr>
<th>Dec Code</th>
<th>Codice Hex</th>
<th>Mnemonico</th>
<th>wire</th>
</tr>
</thead>
<tbody>
<tr>
<td>33 65</td>
<td>21 41</td>
<td>! A</td>
<td>ASCII(wire)</td>
</tr>
</tbody>
</table>

This command will perform the following operations on the 1-Wire® BUS line specified by parameter `wire`.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receive next command.

b) If the device is present, the **Alarm serch** command (code **EC Hex** of 1-Wire® BUS protocol) is sent.

The value of `wire` must be 0 or 1, or the command is ignored.

**Answer Codes:**
The response to this command can be:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Device 1-Wire® present with alarm flag not set</td>
</tr>
<tr>
<td>1</td>
<td>Device 1-Wire® present with alarm flag set</td>
</tr>
<tr>
<td>7</td>
<td>Device 1-Wire® not present and command not sent</td>
</tr>
</tbody>
</table>

**Example:**
If the User wants to send the Alarm serch command to 1-Wire® BUS number 0, then he/she will have to send the following sequence: 33 65 0
9 BITS MASTER-SLAVE COMMUNICATION MODE

The Master-Slave communication takes advantage of the 9 bits mode. This means that a ninth bit is used to distinguish between a call from a "Master" device to one of the "Slave" structures and a mere communication of datas between the Master and the selected Slave. When the ninth bit is set to 1, the data byte must contain the name, or ID code, of the new target device, while if the ninth bit is set to 0 it is possible to send or receive informations from the selected target device.

If the communication is running under ALB protocol the ID code must be the byte set in SETUP mode (NAME). When this byte is received by a device (with the ninth bit set to 1) it recognizes itself and starts to wait for a string containing datas or commands (with the ninth bit set to 0); the string must be maximum 24 bytes long.

It can contain only one command which requires to return an answer code through the serial line, more commands of this kind will be ignored.

The delay between two consecutive characters must be lower than Time-Out, because otherwise the string is considered terminated and the answering phase starts.

Here follows the list of Time-Outs related to the Baud Rate:

<table>
<thead>
<tr>
<th>Baud Rate</th>
<th>Time-Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>38400 Baud</td>
<td>550 μsec</td>
</tr>
<tr>
<td>19200 Baud</td>
<td>990 μsec</td>
</tr>
<tr>
<td>9600 Baud</td>
<td>1.54 msec</td>
</tr>
<tr>
<td>4800 Baud</td>
<td>3.08 msec</td>
</tr>
<tr>
<td>2400 Baud</td>
<td>6.105 msec</td>
</tr>
<tr>
<td>1200 Baud</td>
<td>12.1 msec</td>
</tr>
</tbody>
</table>

When the Time-Out happens, the answer sequence begins; this is made of a byte containing the presence code 6 (6 Hex), or a data sequence requested by a read command sent during the previous call.

Example:
If a string containing the Port read command is sent, the answer to that call will be the presence code, while the answer to the next call will be the data acquired by the Port sent in the previous request.

After having sent the last character of the string the user will have to wait for a time:

"One char transmission time" + Time-Out

before receiving the first char of the answer sequence.

Example:
At 38.4 KBaud, After having sent the last character of the string the User will have to wait for about 810 μsec before to receive the first char of the answer sequence.
NOTE:

1) Between two calls it is essential to wait for a time that depends on the number of commands sent and the type of operations that they involve to be sure that the transmitted command will be executed correctly.

2) If the master unit is not capable to transfer 9 bits data, it is possible to simulate this kind of communication through the parity bit and programming opportunaly the parity even or odd before transmitting each single byte.

The Byte to transmit has an EVEN number of bits 1
 IF Bit 9 is required to be 1 -> Set parity to ODD
 IF Bit 9 is required to be 0 -> Set parity to EVEN

The Byte to transmit has an ODD number of bits 1
 IF Bit 9 is required to be 1 -> Set parity to EVEN
 IF Bit 9 is required to be 0 -> Set parity to ODD
FIGURE 49: POSSIBLE CONNECTIONS DIAGRAM

POWER SUPPLY
- 11 Bits Analog input:
  +10 Vdc
  or
  +8 Vdc
- 9 INPUT LINES OPTO COUPLED
- 4 OUTPUT LINES RELAYS or TRANSISTORS
- 2 Generic TTL I/O lines

POWER SUPPLY
- +10 Vdc or +8 Vdc

ANALOG INPUT
- 0 to 10,000 V
- 4 to 20 mA

DIN 46277-1 and DIN 46277-3 OMEGA RAILS

PLC
- Hardware Serial Line TTL, RS-232, RS-422, RS-485 or Current Loop

QTP terminals

PC like or Macintosh

GPC ® R/T94

GPC ® R/T63

ITALIAN TECHNOLOGY
BIBLIOGRAPHY

In this chapter there is a complete list of technical books, where the user can find all the necessary documentations on the components mounted on GPC® R/T94.

Manual TEXAS INSTRUMENTS: The TTL Data Book - SN54/74 Families
Manual TEXAS INSTRUMENTS: RS-422 and RS-485 Interface Circuits
Manual MAXIM: New Releases Data Book - Volume 4
Manual XICOR: Data Book
Manual PHILIPS: IC12 - F/C bus
Manual ATMEL: Microcontroller - AT89 series
Manual TOSHIBA: Photo Couplers - Data Book
Manual MOTOROLA: Bipolar Power Transistor Data
Manuale HEWLETT PACKARD: Optoelectronics Designer's Catalog
Technical notes: LM2825N
DALLAS technical notes: Documentation and application notes about 1-Wire® communication protocol and devices

For further information and upgrades please refer to specific internet web pages of the manufacturing companies.
Here follow the summarizing tables of firmware ALB x94 commands.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>DEC. Code</th>
<th>HEX Code</th>
<th>MNEMONIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master reset</td>
<td>65 97</td>
<td>41 61</td>
<td>A a</td>
</tr>
<tr>
<td>Read firmware version</td>
<td>86</td>
<td>56</td>
<td>V</td>
</tr>
<tr>
<td>Check EEPROM ready for writing</td>
<td>66</td>
<td>42</td>
<td>B</td>
</tr>
<tr>
<td>Presence byte input</td>
<td>121</td>
<td>79</td>
<td>y</td>
</tr>
<tr>
<td>AUX signal configuration setting</td>
<td>85 cfg</td>
<td>55 cfg</td>
<td>U ASCII(cfg)</td>
</tr>
<tr>
<td>AUX signal configuration acquisition</td>
<td>102</td>
<td>66</td>
<td>f</td>
</tr>
<tr>
<td>Output port set</td>
<td>87 1 dat 0</td>
<td>57 1 dat 0</td>
<td>W SOH ASCII(dat) NUL</td>
</tr>
<tr>
<td>Input port acquisition</td>
<td>82 0 dat 0</td>
<td>52 0 dat 0</td>
<td>R NUL</td>
</tr>
<tr>
<td>Output bit set</td>
<td>83 1 bit</td>
<td>53 1 bit</td>
<td>S SOH ASCII(bit)</td>
</tr>
<tr>
<td>Timed output bit set</td>
<td>115 1 tmp.L tmp.H</td>
<td>73 1 tmp.L tmp.H</td>
<td>s SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)</td>
</tr>
<tr>
<td>Output bit clear</td>
<td>67 1 bit</td>
<td>43 1 bit</td>
<td>C SOH ASCII(bit)</td>
</tr>
<tr>
<td>Timed output bit clear</td>
<td>99 1 tmp.L tmp.H</td>
<td>63 1 tmp.L tmp.H</td>
<td>c SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)</td>
</tr>
<tr>
<td>Input bit or AUX input acquisition</td>
<td>114 port bit</td>
<td>72 port bit</td>
<td>r ASCII(port) ASCII(bit)</td>
</tr>
<tr>
<td>Square wave output bit</td>
<td>80 1 tmp.L tmp.H</td>
<td>50 1 tmp.L tmp.H</td>
<td>P SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)</td>
</tr>
<tr>
<td>16 bit counter read</td>
<td>73</td>
<td>49</td>
<td>I</td>
</tr>
<tr>
<td>16 bit counter reset</td>
<td>88 120</td>
<td>58 78</td>
<td>X x</td>
</tr>
</tbody>
</table>

**Figure A1: ALB x94 commands summarizing table 1**
<table>
<thead>
<tr>
<th>COMMAND</th>
<th>DEC. Code</th>
<th>HEX Code</th>
<th>MNEMONIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last memorizable message acqisition</td>
<td>77</td>
<td>4D</td>
<td>M</td>
</tr>
<tr>
<td>Storing a message</td>
<td>69</td>
<td>45</td>
<td>ASCII(msg.L) ASCII(msg.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(car0.L) ASCII(car0.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(car9.L) ASCII(car9.H)</td>
</tr>
<tr>
<td>Reading a message</td>
<td>79</td>
<td>4C</td>
<td>ASCII(msg.L) ASCII(msg.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(car0.L) ASCII(car0.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(car9.L) ASCII(car9.H)</td>
</tr>
<tr>
<td>Clock setting</td>
<td>79</td>
<td>4F</td>
<td>ASCII(ore.L) ASCII(ore.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(min.L) ASCII(min.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(sec.L) ASCII(sec.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(gio.L) ASCII(gio.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(mes.L) ASCII(mes.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(ann.L) ASCII(ann.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(gse.L) ASCII(gse.H)</td>
</tr>
<tr>
<td>Clock read</td>
<td>111</td>
<td>6F</td>
<td>O</td>
</tr>
<tr>
<td>RTC SRAM write</td>
<td>71</td>
<td>47</td>
<td>ASCII(ind.L) ASCII(ind.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(dat.L) ASCII(dat.H)</td>
</tr>
<tr>
<td>RTC SRAM read</td>
<td>103</td>
<td>67</td>
<td>ASCII(ind.L) ASCII(ind.H)</td>
</tr>
<tr>
<td>Reset BUS 1-Wire®</td>
<td>33 85</td>
<td>21 55</td>
<td>U ASCII(wire)</td>
</tr>
<tr>
<td>WR bit BUS 1-Wire®</td>
<td>33 119</td>
<td>21 77</td>
<td>w ASCII(wire) ASCII(bit)</td>
</tr>
<tr>
<td>RD bit BUS 1-Wire®</td>
<td>33 114</td>
<td>33 72</td>
<td>r ASCII(wire)</td>
</tr>
<tr>
<td>Write n bytes to BUS 1-Wire®</td>
<td>33 87</td>
<td>21 57</td>
<td>! W ASCII(wire) ASCII(n)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(dat.L) ASCII(dat.H)</td>
</tr>
<tr>
<td>Read n bytes from BUS 1-Wire®</td>
<td>33 82</td>
<td>21 52</td>
<td>! R ASCII(wire) ASCII(n)</td>
</tr>
<tr>
<td>Read ROM (1-Wire®)</td>
<td>33 76</td>
<td>33 4C</td>
<td>! L ASCII(wire)</td>
</tr>
<tr>
<td>Match ROM (1-Wire®)</td>
<td>33 77</td>
<td>33 4D</td>
<td>! M ASCII(wire)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(rom0.L) ASCII(rom0.H)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ASCII(rom7.L) ASCII(rom7.H)</td>
</tr>
<tr>
<td>Skip ROM (1-Wire®)</td>
<td>33 83</td>
<td>33 53</td>
<td>! S ASCII(wire)</td>
</tr>
<tr>
<td>Alarm search (1-Wire®)</td>
<td>33 65</td>
<td>33 41</td>
<td>! A ASCII(wire)</td>
</tr>
</tbody>
</table>

**Figure A2: ALB x94 commands summarizing table 2**
APPENDIX B: ON BOARD COMPONENTS DESCRIPTION

Features
- Compatible with MCS-51™ Products
- 4K Bytes of Reprogrammable Flash Memory
  - Endurance: 1,000 Write/Erase Cycles
- 3.0V to 6V Operating Range
- Fully Static Operation: 0 Hz to 24 MHz
- Two-Level Program Memory Lock
- 128 x 8-Bit Internal RAM
- 15 Programmable I/O Lines
- Two 16-Bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial UART Channel
- Direct LED Drive Outputs
- On-Chip Analog Comparator
- Low Power Idle and Power Down Modes
- Brown-Out Detection

Description
The AT89C4051 is a low-voltage, high-performance CMOS 8-bit microcomputer with 4K Bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Atmel’s high density nonvolatile memory technology and is compatible with the industry standard MCS-51™ instruction set. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C4051 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

The AT89C4051 provides the following standard features: 4K Bytes of Flash, 128 bytes of RAM, 15 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, a precision analog comparator, on-chip oscillator and clock circuitry. In addition, the AT89C4051 is designed with static logic for operation down to zero frequency and supports two software-selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The Power Down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

Pin Configuration

<table>
<thead>
<tr>
<th>PDIP/SOIC</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>RST/VPP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(RXD)</td>
<td>2</td>
<td>19</td>
<td>18</td>
<td>17</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>(TXD)</td>
<td>3</td>
<td>18</td>
<td>17</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XTAL2</td>
<td>4</td>
<td>17</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XTAL1</td>
<td>5</td>
<td>16</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(INTO)</td>
<td>6</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(INT1)</td>
<td>7</td>
<td>14</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(T0)</td>
<td>8</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(T1)</td>
<td>9</td>
<td>12</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GND</td>
<td>10</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

VCC
Pin Description

Vcc
Supply voltage.

GND
Ground.

Port 1
Port 1 is an 8-bit bidirectional I/O port. Port pins P1.2 to P1.7 provide internal pullups. P1.0 and P1.1 require external pullups. P1.0 and P1.1 also serve as the positive input (AIN0) and the negative input (AIN1), respectively, of the on-chip precision analog comparator. Port 1 output buffers can sink 20 mA and can drive LED displays directly. When 1s are written to Port 1 pins, they can be used as inputs. When pins P1.2 to P1.7 are used as inputs and are externally pulled low, they will source current (IIL) because of the internal pullups. Port 1 also receives code data during Flash programming and verification.

Port 3
Port 3 pins P3.0 to P3.5, P3.7 are seven bidirectional I/O pins with internal pullups. P3.6 is hard-wired as an input to the output of the on-chip comparator and is not accessible as a general purpose I/O pin. The Port 3 output buffers can sink 20 mA. When 1s are written to Port 3 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pullups. Port 3 also serves the functions of various special features of the AT89C4051 as listed below:

- Port Pin Alternate Functions
  - P3.0 RXD (serial input port)
  - P3.1 TXD (serial output port)
  - P3.2 INT0 (external interrupt 0)
  - P3.3 INT1 (external interrupt 1)
  - P3.4 T0 (timer 0 external input)
  - P3.5 T1 (timer 1 external input)

Port 3 also receives some control signals for Flash programming and verification.

RST
Reset input. All I/O pins are reset to 1s as soon as RST goes high. Holding the RST pin high for two machine cycles while the oscillator is running resets the device. Each machine cycle takes 12 oscillator or clock cycles.

XTAL2
Output from the inverting oscillator amplifier.

Oscillator Characteristics
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 2.

There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 1. Oscillator Connections

Note: C1, C2 = 30 pF ± 10 pF for Crystals
      = 40 pF ± 10 pF for Ceramic Resonators

Figure 2. External Clock Drive Configuration
Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in the table below. Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

Table 1. AT89C4051 SFR Map and Reset Values

<table>
<thead>
<tr>
<th>Address</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>ACC</td>
</tr>
<tr>
<td>06H</td>
<td>IP</td>
</tr>
<tr>
<td>08H</td>
<td>P1</td>
</tr>
<tr>
<td>0A8H</td>
<td>IE</td>
</tr>
<tr>
<td>0A0H</td>
<td>IP</td>
</tr>
<tr>
<td>0B8H</td>
<td>EB</td>
</tr>
<tr>
<td>0C8H</td>
<td>SCON</td>
</tr>
<tr>
<td>0C0H</td>
<td>IE</td>
</tr>
<tr>
<td>0C8H</td>
<td>SCON</td>
</tr>
<tr>
<td>0D8H</td>
<td>PSW</td>
</tr>
<tr>
<td>0E8H</td>
<td>IE</td>
</tr>
<tr>
<td>0F8H</td>
<td>IE</td>
</tr>
</tbody>
</table>

User software should not write 1s to these unlisted locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.

Restrictions on Certain Instructions

The AT89C4051 is an economical and cost-effective member of Atmel's growing family of microcontrollers. It contains 4 Kbytes of flash program memory. It is fully compatible with the MCS-51 architecture, and can be programmed using the MCS-51 instruction set. However, there are a few considerations one must keep in mind when utilizing certain instructions to program this device.

At all the instructions related to jumping or branching should be restricted such that the destination address falls within the physical program memory space of the device, which is 4 Kbytes for the AT89C4051. This should be the responsibility of the software programmer. For example, LJMP 0FE0H would be a valid instruction for the AT89C4051 (with 4K of memory), whereas LJMP 1000H would not.

Note: 1. The Lock Bits can only be erased with the Chip Erase operation.

Program Memory Lock Bits

On the chip are two lock bits which can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the table below:

Lock Bit Protection Modes

<table>
<thead>
<tr>
<th>Protection Type</th>
<th>LB1</th>
<th>LB2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No program lock features.</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>Further programming of the Flash is disabled.</td>
<td>P</td>
<td>U</td>
</tr>
<tr>
<td>Same as mode 2, also verify is disabled.</td>
<td>P</td>
<td>P</td>
</tr>
</tbody>
</table>

Note: 1. The Lock Bits can only be erased with the Chip Erase operation.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset. P1.0 and P1.1 should be set to '0' if no external pullups are used, or set to '1' if external pullups are used. It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when idle is terminated by reset, the instruction following the one that invokes idle should not be one that writes to a port pin or to external memory.

Power Down Mode

In the power down mode the oscillator is stopped, and the instruction that invokes power down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power down mode is terminated. The only exit from power down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before VCC is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize. P1.0 and P1.1 should be set to '0' if no external pullups are used, or set to '1' if external pullups are used.
MEMORY ORGANIZATION

Program Memory

The 80C51 has separate address spaces for program and data memory. The Program memory can be up to 64k bytes long. The lower 4k can reside on-chip. Figure 1 shows a map of the 80C51 program memory.

The 80C51 can address up to 64k bytes of data memory to the chip. The MOVX instruction is used to access the external data memory.

The 80C51 has 128 bytes of on-chip RAM plus a number of Special Function Registers (SFRs). The lower 128 bytes of RAM can be accessed either by direct addressing (MOV data addr) or by indirect addressing (MOV @Ri). Figure 2 shows the Data Memory organization.

Direct and Indirect Address Area

The 128 bytes of RAM which can be accessed by both direct and indirect addressing can be divided into three segments as listed below and shown in Figure 3:

1. Register Banks 0-3: Locations 0 through 1FH (32 bytes). The device after reset defaults to register bank 0. To use the other register banks, the user must select them in software. Each register bank contains eight 1-byte registers 0 through 7. Reset initializes the stack pointer to location 07H, and it is incremented once to start from location 08H, which is the first register (R0) of the second register bank. Thus, in order to use more than one register bank, the SP should be initialized to a different location of the RAM, and the new bank is not used for data storage (i.e., the higher part of the RAM).

2. Bit Addressable Area: 16 bytes have been assigned for this segment, 20H-2FH. Each one of the 128 bits of this segment can be directly addressed (0-7FH). The bits can be referred to in two ways, both of which are acceptable by most assemblers. One way is to refer to their address (i.e., 0-7FH). The other way is with reference to bytes 20H to 27H. Thus, bits 0-7 can also be referred to as bits 20.0-20.7, and bits 28H-2FH are the same as 21.0-21.7, and so on. Each of the 16 bytes in this segment can also be addressed as a byte.

3. Scratch Pad Area: 30H through 7FH are available to the user as data RAM. However, if the stack pointer has been initialized to this area, enough bytes should be left aside to prevent SP data destruction.

Figure 2 shows the different segments of the on-chip RAM.

DIRECT ADDRESSING

INTERNAL

EXTERNAL

DRIECT AND INDIRECT ADDRESSING

RAM

64k BYTES

EXTERNAL

0FFF

0000

Figure 1. 80C51 Program Memory

Figure 2. 80C51 Data Memory

Figure 3. 128 Bytes of RAM Direct and Indirect Addressable
# 80C51 Special Function Registers

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>DESCRIPTION</th>
<th>DIRECT ADDRESS</th>
<th>MSB BIT ADDRESS</th>
<th>LSB</th>
<th>RESET VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACC*</td>
<td>Accumulator</td>
<td>0E0H</td>
<td>E7 E6 E5 E4 E3 E2 E1 E0</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>B*</td>
<td>Register</td>
<td>0F0H</td>
<td>F7 F6 F5 F4 F3 F2 F1 F0</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>DPTR</td>
<td>Data pointer</td>
<td>0B0H</td>
<td>B7 B6 B5 B4 B3 B2 B1 B0</td>
<td></td>
<td>00000000B</td>
</tr>
<tr>
<td>DPH</td>
<td>Data pointer high</td>
<td>0B0H</td>
<td></td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>DPL</td>
<td>Data pointer low</td>
<td>0B0H</td>
<td></td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>IE*</td>
<td>Interrupt enable</td>
<td>A8H</td>
<td>A7 A6 A5 A4 A3 A2 A1 A0</td>
<td></td>
<td>0000000B</td>
</tr>
<tr>
<td>IP*</td>
<td>Interrupt priority</td>
<td>B8H</td>
<td>B7 B6 B5 B4 B3 B2 B1 B0</td>
<td></td>
<td>RFH</td>
</tr>
<tr>
<td>P0*</td>
<td>Port 0</td>
<td>0A0H</td>
<td>A9 A8 A7 A6 A5 A4 A3 A2 A1 A0</td>
<td></td>
<td>RFH</td>
</tr>
<tr>
<td>P1*</td>
<td>Port 1</td>
<td>090H</td>
<td></td>
<td></td>
<td>RFH</td>
</tr>
<tr>
<td>P2*</td>
<td>Port 2</td>
<td>0A0H</td>
<td>A7 A6 A5 A4 A3 A2 A1 A0</td>
<td></td>
<td>RFH</td>
</tr>
<tr>
<td>P3*</td>
<td>Port 3</td>
<td>0B0H</td>
<td>T2 T1 T0 T9 T8 T7 T6 T5</td>
<td></td>
<td>RFH</td>
</tr>
<tr>
<td>PCON*</td>
<td>Power control</td>
<td>087H</td>
<td></td>
<td></td>
<td>0000000B</td>
</tr>
<tr>
<td>PSW*</td>
<td>Program status word</td>
<td>0D0H</td>
<td>C7 C6 C5 C4 C3 C2 C1 C0</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>SBUF</td>
<td>Serial data buffer</td>
<td>090H</td>
<td>F9 F8 F7 F6 F5 F4 F3 F2 F1 F0</td>
<td></td>
<td>xxxxxxxxxB</td>
</tr>
<tr>
<td>SCON*</td>
<td>Serial controller</td>
<td>090H</td>
<td>S8 S7 S6 S5 S4 S3 S2 S1 S0</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>SP</td>
<td>Stack pointer</td>
<td>081H</td>
<td></td>
<td></td>
<td>07H</td>
</tr>
<tr>
<td>TCON*</td>
<td>Timer control</td>
<td>086H</td>
<td>T8 T7 T6 T5 T4 T3 T2 T1 T0</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>TH0</td>
<td>Timer high/low</td>
<td>08CH</td>
<td>T9 T8 T7 T6 T5 T4 T3 T2</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>TH1</td>
<td>Timer high/low</td>
<td>08CH</td>
<td>T9 T8 T7 T6 T5 T4 T3 T2</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>TL0</td>
<td>Timer low</td>
<td>08AH</td>
<td>T9 T8 T7 T6 T5 T4 T3 T2 T1 T0</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>TL1</td>
<td>Timer low</td>
<td>08AH</td>
<td>T9 T8 T7 T6 T5 T4 T3 T2 T1 T0</td>
<td></td>
<td>00H</td>
</tr>
<tr>
<td>TMOD</td>
<td>Timer mode</td>
<td>08AH</td>
<td>M8 M7 M6 M5 M4 M3 M2 M1 M0</td>
<td></td>
<td>00H</td>
</tr>
</tbody>
</table>

## Notes:

- * Bit addressable
- ** Bits GF1, GF0, PD, and IDL of the PCON register are not implemented on the NMOS 8051/8031.
Those SFRs that have their bits assigned for various functions are listed in this section. A brief description of each bit is provided for quick reference. For more detailed information refer to the Architecture Chapter of this book.

**PSW: PROGRAM STATUS WORD. BIT ADDRESSABLE.**

<table>
<thead>
<tr>
<th>CY</th>
<th>AC</th>
<th>F0</th>
<th>RS1</th>
<th>RS0</th>
<th>OV</th>
<th>–</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>CY</td>
<td>PSM.7 Carry Flag.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AC</td>
<td>PSM.6 Auxiliary Carry Flag.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F0</td>
<td>PSM.5 Flag 0 available to the user for general purpose.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RS1</td>
<td>PSM.4 Register Bank selector bit 1 (SEE NOTE 1).</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RS0</td>
<td>PSM.3 Register Bank selector bit 0 (SEE NOTE 1).</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OV</td>
<td>PSM.2 Overflow Flag.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>–</td>
<td>PSM.1 Usable as a general purpose flag.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>PSM.0 Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of '1' bits in the accumulator.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:**
1. The value presented by RS0 and RS1 selects the corresponding register bank.

**PCON: POWER CONTROL REGISTER. NOT BIT ADDRESSABLE.**

<table>
<thead>
<tr>
<th>SMOD</th>
<th>–</th>
<th>–</th>
<th>GF1</th>
<th>GFO</th>
<th>PD</th>
<th>IDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMOD</td>
<td>Double baud rate bit. If Timer 1 is used to generate baud rate and SMOD = 1, the baud rate is doubled when the Serial Port is enabled.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>–</td>
<td>Not implemented, reserved for future use.*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>–</td>
<td>Not implemented, reserved for future use.*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>–</td>
<td>Not implemented, reserved for future use.*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>–</td>
<td>General purpose flag bit.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GFO</td>
<td>General purpose flag bit.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PD</td>
<td>Power Down bit. Setting this bit activates Power Down operation in the 80C51. (Available only in CMOS.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDL</td>
<td>Idle mode bit. Setting this bit activates Idle Mode operation in the 80C51. (Available only in CMOS.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If 1's are written to PD and IDL at the same time, PD takes precedence.

* User software should not write 1's to reserved bits. These bits may be used in future 8051 products to invoke new features.

**INTERRUPTS:** To use any of the interrupts in the 80C51 Family, the following three steps must be taken.

1. Set the EA (enable all) bit in the IE register to 1.
2. Set the corresponding individual interrupt enable bit in the IE register to 1.
3. Begin the interrupt service routine at the corresponding Vector Address of that interrupt. See Table below.

**NOTE:**
In addition, for external interrupts, pins INT0 and INT1 (P3.2 and P3.3) must be set to 1, and depending on whether the interrupt is level or transition activated, bits IT0 or IT1 in the TCON register may need to be set to 1.

**IE: INTERRUPT ENABLE REGISTER. BIT ADDRESSABLE.**

If the bit is 0, the corresponding interrupt is disabled. If the bits is 1, the corresponding interrupt is enabled.

**TABLE:**

<table>
<thead>
<tr>
<th>INTERRUPT SOURCE</th>
<th>VECTOR ADDRESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE0</td>
<td>0003H</td>
</tr>
<tr>
<td>TF0</td>
<td>000BH</td>
</tr>
<tr>
<td>IE1</td>
<td>0013H</td>
</tr>
<tr>
<td>TF1</td>
<td>001BH</td>
</tr>
<tr>
<td>RX &amp; TI</td>
<td>0023H</td>
</tr>
</tbody>
</table>

**INTERRUPT SOURCE**

- **EA**: Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
- **ES**: Enable or disable the serial port interrupt.
- **ET1**: Enable or disable the Timer 1 overflow interrupt.
- **EX1**: Enable or disable External Interrupt 1.
- **ET0**: Enable or disable the Timer 0 overflow interrupt.
- **EX0**: Enable or disable External Interrupt 0.

* User software should not write 1's to reserved bits. These bits may be used in future 80C51 products to invoke new features.
ASSIGNING HIGHER PRIORITY TO ONE OR MORE INTERRUPTS:

In order to assign higher priority to an interrupt the corresponding bit in the IP register must be set to 1.
Remember that while an interrupt service is in progress, it cannot be interrupted by a lower or same level interrupt.

PRIORITY WITHIN LEVEL:
Priority within level is only to resolve simultaneous requests of the same priority level.
From high to low, interrupt sources are listed below:
IE0
TF0
IE1
TF1
RI or TI

IP: INTERRUPT PRIORITY REGISTER. BIT ADDRESSABLE.
If the bit is 0, the corresponding interrupt has a lower priority and if the bit is 1 the corresponding interrupt has a higher priority.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP7</td>
<td>Not implemented, reserved for future use.*</td>
</tr>
<tr>
<td>IP6</td>
<td>Not implemented, reserved for future use.*</td>
</tr>
<tr>
<td>IP5</td>
<td>Not implemented, reserved for future use.*</td>
</tr>
<tr>
<td>IP4</td>
<td>Defines the Serial Port interrupt priority level.</td>
</tr>
<tr>
<td>IP3</td>
<td>Defines the Timer 1 interrupt priority level.</td>
</tr>
<tr>
<td>IP2</td>
<td>Defines External Interrupt 1 priority level.</td>
</tr>
<tr>
<td>IP1</td>
<td>Defines the timer 0 interrupt priority level.</td>
</tr>
<tr>
<td>IP0</td>
<td>Defines the External Interrupt 0 priority level.</td>
</tr>
</tbody>
</table>

* User software should not write 1s to reserved bits. These bits may be used in future 8051 products to invoke new features.

TCON: TIMER/COUNTER CONTROL REGISTER. BIT ADDRESSABLE.

- TCON.7 Timer 1 overflow flag. Set by hardware when the Timer/Counter 1 overflows. Cleared by hardware as processor vectors to the interrupt service routine.
- TCON.6 Timer 0 run control bit. Set/cleared by software to turn Timer/Counter 0 ON/OFF.
- TCON.5 Timer 0 overflow flag. Set by hardware when the Timer/Counter 0 overflows. Cleared by hardware as processor vectors to the service routine.
- TCON.3 External Interrupt 1 edge flag. Set by hardware when External interrupt edge is detected. Cleared by hardware when interrupt is processed.
- TCON.2 Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered External interrupt.
- TCON.1 External Interrupt 0 edge flag. Set by hardware when External interrupt edge detected. Cleared by hardware when interrupt is processed.
- TCON.0 Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered External interrupt.

TMOD: TIMER/COUNTER MODE CONTROL REGISTER. NOT BIT ADDRESSABLE.

- GATE When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control).
  When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control).
- C/T Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin).
- M1 Mode selector bit. (NOTE 1)
- M0 Mode selector bit. (NOTE 1)

NOTE 1:
- M1 M0 Operating Mode
- 0 0 13-bit Timer (8048 compatible)
- 0 1 16-bit Timer Counter
- 1 0 2 8-bit Auto-Reload Timer Counter
- 1 1 3 (Timer 0): T0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. T0 is auto Reload Timer and is controlled by Timer 1 control bits.
- 0 1 3 (Timer 1): Timer/Counter 1 stopped
### TIMER SET-UP

Tables 2 through 5 give some values for TMOD which can be used to set up Timer 0 in different modes. It is assumed that only one timer is being used at a time. If it is desired to run Timers 0 and 1 simultaneously, in any mode, the value in TMOD for Timer 0 must be ORed with the value shown for Timer 1 (Tables 5 and 6).

For example, if it is desired to run Timer 0 in mode 1 GATE (external control) and Timer 1 in mode 2 COUNTER, then the value that must be loaded into TMOD is 69H (09H from Table 2 ORed with 60H from Table 5).

Moreover, it is assumed that the user at this point is not ready to turn the timers on and will do that at a different point in the program by setting the bit TRx (in TCON) to 1.

#### TIMER/COUNTER 0

**Table 2. As a Timer:**

<table>
<thead>
<tr>
<th>MODE</th>
<th>TMOD</th>
<th>FUNCTION</th>
<th>INTERNAL CONTROL</th>
<th>EXTERNAL CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00H</td>
<td>13-bit Timer</td>
<td>00H</td>
<td>00H</td>
</tr>
<tr>
<td>1</td>
<td>09H</td>
<td>16-bit Timer</td>
<td>09H</td>
<td>09H</td>
</tr>
<tr>
<td>2</td>
<td>0AH</td>
<td>8-bit Auto-Reload</td>
<td>0AH</td>
<td>0AH</td>
</tr>
<tr>
<td>3</td>
<td>0BH</td>
<td>Two 8-bit Timers</td>
<td>0BH</td>
<td>0BH</td>
</tr>
</tbody>
</table>

**Table 3. As a Counter:**

<table>
<thead>
<tr>
<th>MODE</th>
<th>TMOD</th>
<th>FUNCTION</th>
<th>INTERNAL CONTROL</th>
<th>EXTERNAL CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01H</td>
<td>13-bit Timer</td>
<td>01H</td>
<td>01H</td>
</tr>
<tr>
<td>1</td>
<td>02H</td>
<td>16-bit Timer</td>
<td>02H</td>
<td>02H</td>
</tr>
<tr>
<td>2</td>
<td>03H</td>
<td>8-bit Auto-Reload</td>
<td>03H</td>
<td>03H</td>
</tr>
<tr>
<td>3</td>
<td>04H</td>
<td>Two 8-bit Timers</td>
<td>04H</td>
<td>04H</td>
</tr>
</tbody>
</table>

**NOTES:**

1. The timer is turned ON/OFF by setting/clearing bit TR0 in the software.
2. The Timer is turned ON/OFF by the 1-to-0 transition on INT0 (P3.2) when TR0 = 1 (hardware control).

#### TIMER/COUNTER 1

**Table 4. As a Timer:**

<table>
<thead>
<tr>
<th>MODE</th>
<th>TMOD</th>
<th>FUNCTION</th>
<th>INTERNAL CONTROL</th>
<th>EXTERNAL CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00H</td>
<td>13-bit Timer</td>
<td>00H</td>
<td>00H</td>
</tr>
<tr>
<td>1</td>
<td>09H</td>
<td>16-bit Timer</td>
<td>09H</td>
<td>09H</td>
</tr>
<tr>
<td>2</td>
<td>0AH</td>
<td>8-bit Auto-Reload</td>
<td>0AH</td>
<td>0AH</td>
</tr>
<tr>
<td>3</td>
<td>0BH</td>
<td>Does not run</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

**Table 5. As a Counter:**

<table>
<thead>
<tr>
<th>MODE</th>
<th>TMOD</th>
<th>FUNCTION</th>
<th>INTERNAL CONTROL</th>
<th>EXTERNAL CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01H</td>
<td>13-bit Timer</td>
<td>01H</td>
<td>01H</td>
</tr>
<tr>
<td>1</td>
<td>02H</td>
<td>16-bit Timer</td>
<td>02H</td>
<td>02H</td>
</tr>
<tr>
<td>2</td>
<td>03H</td>
<td>8-bit Auto-Reload</td>
<td>03H</td>
<td>03H</td>
</tr>
<tr>
<td>3</td>
<td>–</td>
<td>Not available</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

**NOTES:**

1. The timer is turned ON/OFF by setting/clearing bit TR1 in the software.
2. The Timer is turned ON/OFF by the 1-to-0 transition on INT1 (P3.2) when TR1 = 1 (hardware control).
SCON: SERIAL PORT CONTROL REGISTER. BIT ADDRESSABLE.

<table>
<thead>
<tr>
<th>SMD</th>
<th>SM1</th>
<th>SM2</th>
<th>REN</th>
<th>TB8</th>
<th>RB8</th>
<th>TI</th>
<th>RI</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMD</td>
<td>SCON.7</td>
<td>Serial Port mode specifier. (NOTE 1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SM1</td>
<td>SCON.6</td>
<td>Serial Port mode specifier. (NOTE 1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SM2</td>
<td>SCON.5</td>
<td>Enables the multiprocessor communication feature in modes 2 &amp; 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 0, if SM2 = 1 then RI should be 0. (See Table 6.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REN</td>
<td>SCON.4</td>
<td>Set/Cleared by software to Enable/Disable reception.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TB8</td>
<td>SCON.3</td>
<td>The 9th bit that will be transmitted in modes 2 &amp; 3. Set/Clear by software.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB8</td>
<td>SCON.2</td>
<td>In modes 2 &amp; 3, is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TI</td>
<td>SCON.1</td>
<td>Transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes. Must be cleared by software.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RI</td>
<td>SCON.0</td>
<td>Receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes (except see SM2). Must be cleared by software.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NOTE 1:

<table>
<thead>
<tr>
<th>SM0</th>
<th>SM1</th>
<th>Mode</th>
<th>Description</th>
<th>Baud Rate</th>
<th>Osc Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8 bit UART</td>
<td>Variable</td>
<td>FOSC /12</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>9 bit UART</td>
<td>Variable</td>
<td>FOSC /96 or FOSC /32</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>9 bit UART</td>
<td>Variable</td>
<td>FOSC /64 or FOSC /16</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>9 bit UART</td>
<td>Variable</td>
<td>FOSC /12</td>
</tr>
</tbody>
</table>

SERIAL PORT SET-UP:

<table>
<thead>
<tr>
<th>Mode</th>
<th>SCON</th>
<th>SM2 VARIATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Single Processor Environment (SM2 = 0)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Multi-processor Environment (SM2 = 1)</td>
</tr>
</tbody>
</table>

GENERATING BAUD RATES

Serial Port in Mode 0:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Baud Rate</th>
<th>Osc Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/12 Fosc</td>
<td>Fosc /12</td>
</tr>
</tbody>
</table>

Serial Port in Mode 1:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Baud Rate</th>
<th>Osc Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1/32 Fosc</td>
<td>Fosc /32</td>
</tr>
</tbody>
</table>

USING TIMER/COUNTER 1 TO GENERATE BAUD RATES:

For this purpose, Timer 1 is used in mode 2 (Auto-Reload). Refer to Timer Setup section of this chapter.

Baud Rate Osc Freq
32 12 [256 (TH1)]
12 32 [256 (TH1)]

If SMOD = 0, then K = 1.

If SMOD = 1, then K = 2 (SMOD is in the PCON register).

Most of the time the user knows the baud rate and needs to know the reload value for TH1.

TH1 = 256 K Osc Freq
384 baud rate

TH1 must be an integer value. Rounding off TH1 to the nearest integer may not produce the desired baud rate. In this case, the user may have to choose another crystal frequency.

Since the PCON register is not bit addressable, one way to set the bit is logical ORing the PCON register (i.e., ORL PCON,#80H). The address of PCON is 87H.

SERIAL PORT IN MODE 2:

The baud rate is fixed in this mode and is 1/32 or 1/64 of the oscillator frequency, depending on the value of the SMOD bit in the PCON register.

In this mode none of the Timers are used and the clock comes from the internal phase 2 clock.

SMOD = 1, Baud Rate = 1/32 Osc Freq
SMOD = 0, Baud Rate = 1/64 Osc Freq

To set the SMOD bit: ORL PCON,#80H. The address of PCON is 87H.

SERIAL PORT IN MODE 3:

The baud rate in mode 3 is variable and sets up exactly the same as in mode 1.
Table 7. 80C51 Instruction Set Summary (Continued)

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Address Size</th>
<th>Oscillator Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD A,Rn</td>
<td>Add register to Accumulator</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
<tr>
<td>ADD A,direct</td>
<td>Add immediate data to Accumulator</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
<tr>
<td>ADD A,@Ri</td>
<td>Add indirect RAM to Accumulator with carry</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
<tr>
<td>ADD A,Acc</td>
<td>Add immediate data to Accumulator with carry</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
<tr>
<td>SUBB A,direct</td>
<td>Subtract Register from Accumulator with borrow</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
<tr>
<td>SUBB A,@Ri</td>
<td>Subtract indirect RAM from Accumulator with borrow</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
<tr>
<td>INC A</td>
<td>Increment Accumulator</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
<tr>
<td>INC Rn</td>
<td>Increment register</td>
<td>1 byte</td>
<td>12 nanoseconds</td>
</tr>
</tbody>
</table>

Notes on instruction set and addressing modes:

- Direct: 8-bit internal data location's address. This could be an Internal Data RAM location (0-127) or a SFR (i.e., I/O port, control register, status register, etc. (128-255)).
- Indirect: 16-bit constant included in the instruction.
- Address: 16-bit destination address. Used by CALL and JUMP. A branch can be anywhere within the 64k-byte Program Memory address space.
- #data: 8-bit constant included in the instruction.
- @Ri: 8-bit internal data RAM location (0-255) addressed indirectly through register R1 or R0.
- bit: Direct Addressed bit in Internal Data RAM or Special Function Register.

(1) Note that operations on SFR byte address 208 or bit addresses 209-215 (i.e., the PSW or bits in the PSW) also affect flag settings.
### Table 7. 80C51 Instruction Set Summary (Continued)

<table>
<thead>
<tr>
<th>MNEMONIC</th>
<th>DESCRIPTION</th>
<th>BYTE</th>
<th>OSCILLATOR PERIOD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DATA TRANSFER</strong> (Continued)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV A, #data</td>
<td>Move immediate data to Accumulator</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>MOV Rn, A</td>
<td>Move Accumulator to register</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>MOV Rdi, direct</td>
<td>Move direct byte to register</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>MOV Rdi, Rn</td>
<td>Move Accumulator to direct byte</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>MOV direct, Rn</td>
<td>Move register to direct byte</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>MOV direct, direct</td>
<td>Move direct byte to direct</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>MOV direct, @Ri</td>
<td>Move indirect RAM to direct byte</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>MOV direct, #data</td>
<td>Move immediate data to direct byte</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>MOV @Ri, A</td>
<td>Move Accumulator to indirect RAM</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>MOV @Ri, direct</td>
<td>Move direct byte to indirect RAM</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>MOV @Ri, @Ri</td>
<td>Move indirect RAM to indirect RAM</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>MOV DPTR, #data16</td>
<td>Load Data Pointer with a 16-bit constant</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>MOVC A, @A+DPTR</td>
<td>Move Code byte relative to DPTR to A</td>
<td>CC</td>
<td>12</td>
</tr>
<tr>
<td>MOVC A, @A+PC</td>
<td>Move Code byte relative to PC to A</td>
<td>CC</td>
<td>12</td>
</tr>
<tr>
<td>MOVX A, @Ri</td>
<td>Move external RAM (8-bit addr) to A</td>
<td>CC</td>
<td>12</td>
</tr>
<tr>
<td>MOVX A, @DPTR</td>
<td>Move external RAM (16-bit addr) to A</td>
<td>CC</td>
<td>12</td>
</tr>
<tr>
<td>MOVX A, @Ri, A</td>
<td>Move A to external RAM (8-bit addr)</td>
<td>1</td>
<td>24</td>
</tr>
<tr>
<td>MOVX @DPTR, A</td>
<td>Move A to external RAM (16-bit addr)</td>
<td>1</td>
<td>24</td>
</tr>
<tr>
<td>PUSH direct</td>
<td>Push direct byte onto stack</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>POP direct</td>
<td>Pop direct byte from stack</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>XCH A, #direct</td>
<td>Exchange register with Accumulator</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>XCH A, direct</td>
<td>Exchange direct byte with Accumulator</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>XCH A, @Ri</td>
<td>Exchange indirect RAM with Accumulator</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>XCH A, @Ri</td>
<td>Exchange low-order digit indirect RAM with AC(0)</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td><strong>BOOLEAN VARIABLE MANIPULATION</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JB bit, rel</td>
<td>Jump if direct bit is set</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>JNB bit, rel</td>
<td>Jump if direct bit is not set</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>JBC bit, rel</td>
<td>Jump if direct bit is set and set</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td><strong>PROGRAM BRANCHING</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACALL addr11</td>
<td>Absolute subroutine call</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>LCALL addr16</td>
<td>Long subroutine call</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>RET</td>
<td>Return from subroutine</td>
<td>1</td>
<td>24</td>
</tr>
<tr>
<td>RETI</td>
<td>Return from interrupt</td>
<td>1</td>
<td>24</td>
</tr>
<tr>
<td>A JMP @A+DPTR</td>
<td>Jump indirect relative to the DPTR</td>
<td>1</td>
<td>24</td>
</tr>
<tr>
<td>AJMP addr11</td>
<td>Absolute jump</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>L JMP addr16</td>
<td>Long jump</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>SJMP rel</td>
<td>Short jump (relative addr)</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>DJNZ Rn, rel</td>
<td>Decrement register and jump if not zero</td>
<td>2</td>
<td>24</td>
</tr>
<tr>
<td>DJNZ direct, rel</td>
<td>Decrement direct byte and jump if not zero</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>NOP</td>
<td>No operation</td>
<td>1</td>
<td>12</td>
</tr>
</tbody>
</table>

All mnemonics copyrighted © Intel Corporation 1980.
APPENDIX C: ALPHABETICAL INDEX

SYMBOLS
1-WIRE® 7, 53, 69
16 BIT COUNTER READ 64
16 BIT COUNTER RESET 64
9 BITS MASTER-SLAVE COMMUNICATION MODE 75

A
A/D CONVERTER 2, 7, 8, 11, 30, 34, 48
ABACO®LINK BUS 2, 4, 7
ALARM SEARCH COMMAND ON 1-WIRE® BUS, command 74
ALB (ABACO® LINK BUS) SOFTWARE DESCRIPTION 50
ANALOG INPUTS SELECTION 35
AUX SIGNAL CONFIGURATION ACQUISITION, command 56
AUX SIGNAL CONFIGURATION SETTING, command 56

B
BACK UP 2, 32
BACKED SRAM + SERIAL RTC 44
BIBLIOGRAPHY 78

C
CARD VERSION 1
CHECK EEPROM READY FOR WRITING 54
CKS.AMP8 8, 11, 16
CLOCK 6
CLOCK READ 68
CLOCK SETTING 67
COMMANDS TO MANAGE THE 1-WIRE® BUS LINES 70
CONNECTIONS 14
CN3 16
CN4 14
CN6 30
CN7 24
CN8 26, 28
CPU 2, 4, 8, 11
CPU PERIPHERALS 47
CURRENT CONSUMPTION 10
CURRENT LOOP 2, 7, 8, 11, 16, 34, 40

D
DEBOUNCED READ OF AN INPUT PORT BIT, COMMAND, command 60
DIGITAL I/O PORT BIT MANAGEMENT COMMANDS 58
DIGITAL I/O PORT BYTE MANAGEMENT COMMANDS 57
DIGITAL INPUTS 45
DIGITAL OUTPUTS 11, 47

E
ELECTRIC FEATURES OF GPC® R94 10
ELECTRIC FEATURES OF GPC® T94 12

F
FLASH 2, 4

G
GENERAL FEATURES OF GPC® R94 8
GENERAL FEATURES OF GPC® T94 11
GENERAL INFORMATIONS 2

I
I/O CONNECTION 34
I/O TTL LINES 7, 30, 53
INPUT BIT OR AUX INPUT ACQUISITION 60
INPUT PORT ACQUISITION, command 57
INPUT SECTION 7
INSTALLATION 14
INT 2
INTERRUPTS 32
INTRODUCTION 1

J
JUMPER J1 (RUN/SETUP) - A/D LINE 47
JUMPERS 36
  2 PINS JUMPERS 38
  3 AND 5 PINS JUMPERS 36

L
LAST MEMORIZABLE MESSAGE ACQUISITION 65
LEDS 2, 32

M
MASTER RESET, command 54
MASTER-SLAVE COMMUNICATION 75
MATCH ROM COMMAND ON 1-WIRE® BUS, command 73
MEMORY DEVICES 6
MEMORY SELECTION 38
MESSAGES MANAGEMENT COMMANDS 65

N
NPN OPTOCOUPLED DIGITAL INPUTS 8, 11, 45
NPN OUTPUT SIGNALS 34

O
OPTOCOUPLED NPN INPUT SIGNALS 2
OPTOCOUPLED NPN INPUT SIGNALS 24, 34
OUTPUT BIT CLEAR, command 59
OUTPUT BIT SET, command 58
OUTPUT PORT SET, command 57
OUTPUT SECTION 6
OUTPUTS 2

P
PERIPHERAL DEVICES SOFTWARE DESCRIPTION 44
PHYSICAL FEATURES OF GPC® R94 8
PHYSICAL FEATURES OF GPC® T94 11
POWER SUPPLY 14, 16
POWER SUPPLY SECTION 4, 35
PRESENCE BYTE INPUT, command 55
PRESENCE BYTE OUTPUT, command 55

R
READ A BIT FROM 1-WIRE® BUS, command 71
READ BOARD CODE, command 56
READ FIRMWARE VERSION 54
READ ROM CODE FROM 1-WIRE® BUS, command 72
READING A MESSAGE, command 66
REAL TIME CLOCK 2, 6, 8, 11
RELATIVE HUMIDITY 8, 11
RELAY OUTPUTS CONNECTOR 26
RELAYS DIGITAL OUTPUTS 8
RELAYS OUTPUT SIGNALS 34
RESET 1-WIRE® BUS AND GET PRESENCE PULSE, COMMAND 70
RS 232 2, 7, 8, 11, 16, 34, 40
RS 422 2, 7, 8, 11, 16, 34, 40
RS 485 2, 7, 8, 11, 16, 34, 42
RS422-485 TERMINATION NETWORK 10, 12
RTC 44
RTC SRAM READ, command 68
RTC SRAM WRITE, command 68
RUN MODE 53
S

SERIAL COMMUNICATION SELECTION  7, 40
SERIAL EEPROM  2, 6, 8, 11, 38, 44, 55, 66
SETUP MODE  50
SIZE  8, 11
SKIP ROM COMMAND ON 1-WIRE® BUS, command  73
SOFTWARE  43
SQUARE WAVE OUTPUT BIT, command  63
SQUARE WAVE STARTING WITH "0" OUTPUT BIT, command  62
SQUARE WAVE STARTING WITH "1" OUTPUT BIT, command  61
SRAM  2, 4, 6, 8, 11, 32, 38, 44
SRAM+RTC MANAGEMENT COMMANDS  67
STORING A MESSAGE, command  65
SWITCHING POWER SUPPLY  35

T

TECHNICAL FEATURES OF GPC® T94  11
TEMPERATURE RANGE  8, 11
TIMED OUTPUT BIT CLEAR, command  59
TIMED OUTPUT BIT SET, command  58
TIMER COUNTER  2
TRANSISTOR OUTPUTS CONNECTOR  28
TRANSZORB™  2, 35
TTL  2, 7, 8, 11, 16, 34, 40, 42

W

WEIGHT  8, 11
WRITE A BIT TO 1-WIRE® BUS, command  70
WRITE N BYTES TO 1-WIRE® BUS, command  71